UNIVERSIDAD AUSTRAL DE CHILE

FACULTAD DE CIENCIAS DE LA INGENIERÍA

ESCUELA DE INGENIERÍA CIVIL EN INFORMÁTICA

IMPLEMENTACIÓN DE UN SISTEMA DE AUTENTIFICACIÓN DE DATOS BASADO EN WEB SERVICES

TESIS DE GRADO PARA OPTAR AL TÍTULO PROFESIONAL DE INGENIERO CIVIL EN INFORMÁTICA

PATROCINANTE:

ENRIQUE CRESPO

INGENIERO EN INFORMÁTICA.

CO-PATROCINANTE:

MARTIN SOLAR

DOCENTE INSTITUTO DE INFORMÁTICA.

ANGELA MACARENA REYES TRONCOSO

VALDIVIA – CHILE

2004

CREZCA
Servicios
Computacionales
Ltda.

SANTIAGO, 9 de Junio de 2004

DE : ENRIQUE CRESPO FERNÁNDEZ

A : DIRECTORA ESCUELA INGENIERÍA CIVIL EN INFORMÁTICA

MOTIVO:

INFORME TRABAJO DE TITULACIÓN

Nombre Trabajo de Titulación: . IMPLEMENTACION DE UN SISTEMA DE AUTENTIFICACION DE DATOS BASADO EN EBSERVICES

Nombre del Alumno : ANGELA MACARENA REYES TRONCOSO

FUNDAMENTO DE LA NOTA:

La señorita Reyes ha cumplido con creces el objetivo propuesto al inicio de este trabajo. Sus resultados son de aplicación inmediata para la solución del problema de la actualización de la información proveniente de fuentes heterogéneas, numerosas y en las que la legislación sobre la privacidad de la información personal no tiene un canal definido y es un problema abierto.

La forma en la que la señorita Reyes abordó el proyecto y redactó el informe correspondiente, es sin lugar a dudas una clara muestra del rigor científico con el que trabajó.

La comprensión del problema, el acabado conocimiento de las tecnologías subyacentes y el lenguaje utilizado, hacen que la exposición fluya fácilmente aumentando la curiosidad intelectual del lector a medida que transcurre la lectura.

El resumen de las tecnologías disponibles en las actualidad, es en mi opinión, una lectura altamente recomendable a las generaciones nuevas o a aquellas que requieran actualizar sus conocimientos.

La dedicación, disposición y el compromiso demostrados, considero que deben ser destacados.

Probablemente, el único punto débil del trabajo son sus conclusiones, ya que la aplicación de las ideas mostradas en el trabajo a nuestra sociedad y su impacto no fue suficientemente destacada.

Por las razones anteriores, califico este trabajo con nota **6.9 (seis nueve).**

Sin otro particular, le saluda,

Enrique Crespo Fernández
Gerente General
Servicios Computacionales Ltda.

La Concepción 65 • Of. 1001 • FonoFax: 235 0301 • Santiago • Chile • Email: crezca@ctcreuna.cl

Valdivia, 14 de Junio de 2004

De : Martín Gonzalo Solar Monsalves

A : Directora Escuela Ingeniería Civil en Informática

Ref. : Informe Calificación Trabajo de Titulación

Nombre Trabajo de Titulación:

"IMPLEMENTACIÓN DE UN SISTEMA DE AUTENTIFICACIÓN DE DATOS BASADO EN WEB SERVICES"

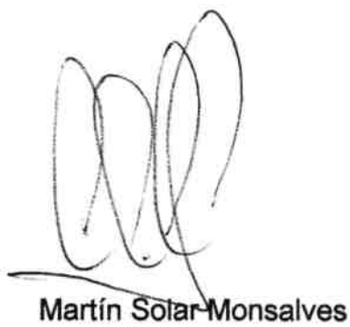
Nombre Alumno:

Ángela Macarena Reyes Troncoso.

Evaluación:

Cumplimiento del objetivo propuesto	6.5
Satisfacción de alguna necesidad	7.0
Aplicación del método científico	6.5
Interpretación de los datos y obtención de conclusiones	6.5
Originalidad	6.0
Aplicación de criterios de análisis y diseño	6.5
Perspectivas del trabajo	6.5
Coherencia y rigurosidad lógica	6.5
Precisión del lenguaje técnico en la exposición, composición, redacción e ilustración	6.5
Nota Final	**6.5**

Sin otro particular, atte.:



Martín Solar Monsalves

Valdivia, 10 Junio del 2 004

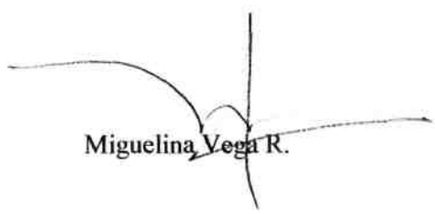
DE: Miguelina Vega Rosales

Profesor Instituto Informática

A: Dirección Escuela Ingeniería Civil en Informática

Informo a usted que el Proyecto de Título " Implementación de un Sistema de Autentificación de Datos Basado en un Web Services", presentado por la señorita Angela Reyes Troncoso, cumple con el objetivo general propuesto, que es formular e implementar un sistema de Servicios Web, que permita entregar a determinar empresas una reducción en los costos de administración en algunos servicios, haciéndolo de manera automática y más accesible, tanto a sus potenciales clientes como a sí misma.

La metodología de trabajo y el lenguaje utilizado es el adecuado, sin embargo, no hace referencia a las pruebas realizadas ni a las mejoras ocasionadas al prototipo implementado. Por lo anteriormente expuesto, califico este proyecto de título con nota 6,2 (seis, dos).



Atentamente

AGRADECIMIENTOS

Agradezco a todas aquellas personas que me apoyaron a terminar este proceso.
A Dios, por haberme dado las oportunidades para llegar a este momento.
En especial a mis amigos... Pao, Víctor y a todos...muchas gracias por todo.
A mis hermanos y Raúl, gracias por todo su apoyo.
Cristian...gracias por toda la ayuda profesional que me brindaste para terminar este trabajo.

Y en especial a mis padres... a ti que eres mi ángel de la guarda, que desde algún lugar del cielo siempre me acompaña... a ti guatón y a mi mamá, dedico esto ...por su cariño, su esfuerzo y su apoyo incondicional... muchas gracias.

Angela.

INDICE

INDICE DE TABLAS

INDICE DE FIGURAS

RESUMEN

Si intentamos centrar el estado actual del desarrollo de aplicaciones basadas en Web, podemos encontrar una gran cantidad de tecnologías, arquitecturas y lenguajes, muchas de ellas incompatibles entre sí. Al revisar la literatura se puede concluir que existe una clara tendencia de las diferentes tecnologías al desarrollo basada en componentes. Existen diferentes plataformas que ofrecen soluciones, todas enmarcadas dentro del modelo de componentes, dentro de ellas los "Web Services" se presenta como una alternativa para facilitar la intercomunicación entre diferentes arquitecturas de componentes y lenguajes, proponiendo una visión de arquitecturas basadas en "servicios".

Como se presenta en este trabajo de titulación, la evolución del desarrollo de aplicaciones web a este tipo de arquitectura permite la integración de diferentes conceptos y protocolos ya conocidos(HTTP, XML), lo cual conlleva a una rápida adopción por parte de las áreas de desarrollo tecnológico en las empresas.

Por otro lado, cada día más empresas están adoptando Internet como un canal de integración empresa-cliente, de acuerdo a lo anterior todos los servicios y/o productos que una empresa antes ofrecía de manera tradicional a sus clientes, ahora están siendo canalizados vía Internet. Dentro de esta evolución, es claro que el manejo de información personal de los clientes es muy importante y obviamente su actualización constante es primordial. Este proceso de ingreso/ validación/actualización/ de datos de los clientes es repetido en cada empresa que necesita el manejo de esta información, lo cual conlleva a hacer repetitivo este proceso a los clientes cada vez que necesitan actualizar su información personal. Este trabajo aborda el diseño e implementación de un prototipo que utilizando las

"bondades" de los "Web Services" ofrece una alternativa de solución para proceso de ingreso/ validación/actualización/ de datos de los clientes.

SUMMARY

If we try to investigate the actual state of the development of the web based applications, we will find a great quantity of different technologies, architectures and programming languages that are often quite incompatible among themselves. After consulting the corresponding literature one can conclude that there is a clear tendency in different technologies towards the component based development. There are many alternative platforms based on the component model. Among them the "Web Services" is presented as one alternative that facilitate the intercommunication between different architectures of the components and languages, bringing one vision of the "service" based architecture.

As presented in this paper, the evolution of the web application development to the Web Services concept permits the integration of different concepts and already well known protocols (HTTP, XML), that implies good reception and adoption by technology development areas in companies.

On the other side, every day there are more companies that are adopting the Internet as a Company-Client integration channel. According to this, all of the services and products that one company was offering in a traditional way, nowadays is channeled via Internet. In this evolution, it is clear that the client's personal information management is very important and it is obvious that its constant actualization is a pressing issue. Client data insertion/validation/actualization process is repeated in each of the companies on the market, what means that this process is redundant and annoying for the clients. This paper discusses the design and implementation of the prototype that using the

"Web Services" offers one solution for the client data insertion/validation/actualization.

CAPÍTULO 1

1. INTRODUCCIÓN

La competitividad entre las diferentes empresas en la actualidad, ha producido una mejora significativa en los servicios ofrecidos a sus distintos clientes. Estos, han mejorado en términos de variedad, facilidad de uso, calidad, utilidad, etc. Ello se ha caracterizado por el acercamiento al cliente mediante la introducción de tecnologías de última generación en los servicios que ofrecen las empresas. Dentro de las más destacadas se mencionan, cajeros automáticos, atención de consultas y la utilización de Internet como nuevo canal de distribución de servicios en línea hacia clientes. [Amo00]

De acuerdo a lo anterior, el aumento de Internet como plataforma de desarrollo de aplicaciones en línea, ha permitido el surgimiento y utilización de diversas tecnologías que presentan características propias de programación. A través de este trabajo se analiza una de las últimas alternativas para el desarrollo y automatización de servicios, especialmente en el área del comercio electrónico; estos son los llamados Web Services[1]

En el presente capítulo se entrega una visión global de lo que propone el modelo de Web Services y sus conceptos básicos, y se revisan los objetivos de este trabajo de titulación.

[1] Servicios Web

1.1 APLICACIONES SOBRE INTERNET.

En la actualidad se observa un punto de vista negativo en el desarrollo de aplicaciones sobre Internet, es el aislamiento que vive cada uno de estos desarrollos. Esto es debido a que pocas veces pensamos en cómo va a comunicarse nuestro portal con el resto de Internet, con el resto de las aplicaciones que necesitamos y que se encuentran en diferentes servidores y escritos en diferentes lenguajes de programación. Esto nos lleva necesariamente a una gran contradicción. En un mundo tan abierto y universal como lo es la red Internet, realmente tenemos un modelo de explotación bastante limitado y atomizado. Un sitio web y otro únicamente se "conocen" por el sistema de hiperenlaces de HTML, y en algunas ocasiones por desarrollos particulares y similares a lo que ahora podremos realizar mediante los servicios web.

Algunos sitios de Internet especializados en la transmisión de información han pensado en este tipo de comunicación o transmisión de datos entre portales, esto debido al tipo de servicio que están ofreciendo, es por esto que habilitan direcciones URL que mediante el paso de una serie de parámetros, devuelven información en distintos formatos (normalmente texto plano y documentos XML).

Si se observa el esquema básico de comunicación en Internet, Figura 1, se verá que ofrecer diferentes servicios sobre un mismo portal resultaría demasiado engorroso en cuanto a tiempo y fiabilida de transacciones.

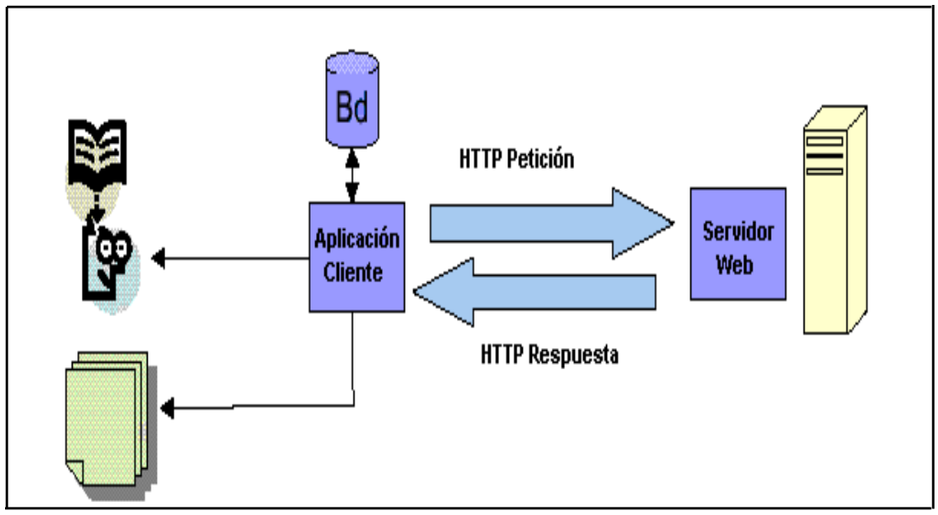


Figura 1, Esquema comunicación Internet

En este trabajo se revisará cómo esta situación ha cambiado paulatinamente. La tecnología Web Services, hace posible desarrollar este tipo de aplicaciones de una forma estandarizada y mucho más sencilla.

La comunicación fluida en las transacciones es la base de un nuevo modelo para ofrecer servicios a través de Internet, que permita realizar una comunicación de módulos de software independiente de toda plataforma.

Este modelo constituye el siguiente paso en la evolución de la tecnología orientada a objetos, y representan una revolución al alejarse de las arquitecturas tradicionales tipo cliente-servidor a nuevas arquitecturas distribuidas tipo peer-to-peer[2]. Los diferentes servicios que son ofrecidos mediante el modelo Web Services permiten crear aplicaciones que utilizan una combinación de módulos de software, lo cual representa una notoria ventaja para el mundo del e-business.

---

[2] Arquitectura de Negocios que permite la transferencia de servicios 1 a 1

Esta es la idea principal de los Web Services, los cuales cumplen con todas las características de calidad, alta disponibilidad y pluralidad que conocemos y requerimos en otro tipo de servicios.

Como se ve en la actualidad los negocios comerciales están provocando una conmoción que sacude los cimientos de los negocios tradicionales. Cada vez son más las empresas que reconocen la oportunidad que presenta Internet y establecen su presencia en la red con un modelo de organización sólida que las respalda. El incremento de los ingresos y de la cantidad de clientes que atrae, ha impulsado a un número importante de empresas ha ofrecer sus productos y servicios en forma digital.

Actualmente Internet no solo ofrece realizar transacciones comerciales sino que, además, permite la incorporación y fusión de diferentes servicios dentro del mismo sistema, esto es idea principal de los Web Services, los que permiten la combinación de diferentes módulos de software que pudiesen estar ubicados en regiones geográficas distintas.

Para esquematizar el funcionamiento de esta tecnología, debemos mencionar que esta compuesto de:

- **Servicio**: corresponde a la o las aplicaciones ofrecidas para ser utilizada por los solicitantes que llenan los servicios especificados por el proveedor de servicios. Tanto la descripción como las políticas de uso deben ser publicadas de antemano en un registro.
- **Proveedor de Servicio**: desde el punto de vista comercial, es quien presta el servicio; desde el punto de vista de arquitectura, es la plataforma que provee el servicio.

- **Registro de servicios**: es un depósito de servicios que puede ser consultado donde los proveedores de servicios publican sus servicios y los solicitantes encuentran los servicios y los detalles para utilizarlos.
- **Solicitante de servicios:** desde el punto de vista comercial es la empresa que quiere cierto servicio; desde el punto de la arquitectura, es la aplicación o cliente que invoca o busca un servicio. [Liz02]

Estos componentes se relacionan de la manera que se observa en la Figura 2.

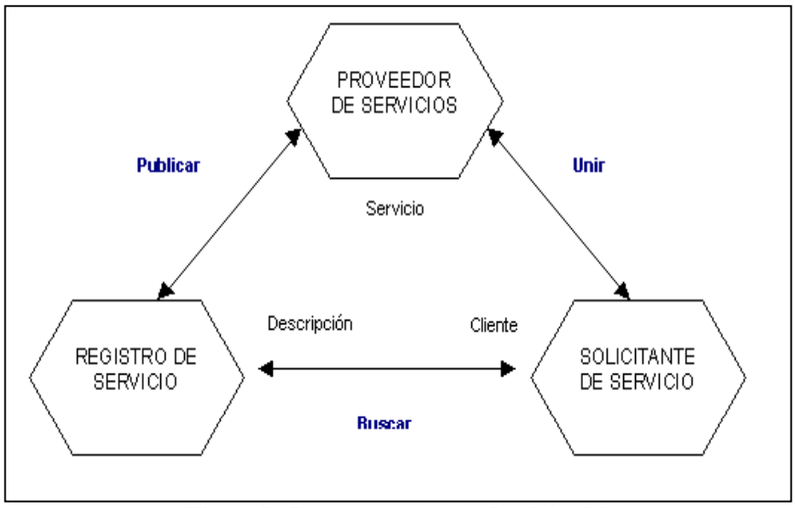


Figura 2, Componentes de los Servicios Web

1.1.1 CARACTERÍSTICAS DE LOS WEB SERVICES.

El principal objetivo que se logra con los Web Services, es la interoperabilidad y la integración. Mediante ellos, las empresas pueden compartir servicios software con sus clientes y sus socios de negocio. Esto ayudará a las compañías a escalar sus negocios, reduciendo el costo en desarrollo y mantenimiento de software, y sacando los productos al mercado con mayor rapidez. La integración de aplicaciones hará posible obtener la información demandada en tiempo real, acelerando el proceso de toma de decisiones.

La evolución de Internet hacia los Web Services, mejorará los resultados globales de las empresas, reduciendo sus gastos y guiándolas hacia una mejora progresiva de la calidad. La adopción de la tecnología de Web Services por la industria es el primer paso hacia una economía global.

Se debe mencionar también que dentro de las cualidades técnicas de este tipo de servicio están:

- **Costo:** con el uso de estándares abiertos, se protege la inversión de la empresa, en cuanto al software que los servicios web emplean, permitiendo un ahorro de costos importante.
- **Facilidad de uso:** con una amplia variedad de herramientas disponibles, esto hace posible que las personas de negocios agreguen diferentes Web Services en los procesos de negocios.
- **Productividad de los desarrolladores:** desde la base de los estándares abiertos y la variedad de herramientas disponibles, los desaprobadores hoy pueden sentir un gran agrado de trabajar con Web Services. [URL 7]

1.1.2 WEB SERVICES Y LOS RIESGOS ASOCIADOS.

Las expectaciones alrededor de esta tecnología son grandes, porque el mercado de aplicación es muy amplio. Pero también tiene sus puntos oscuros. [URL 3]

Los Web Services hacen uso de las mismas tecnologías que han sido atacadas en tantas ocasiones. Usando Web Services, la seguridad de una empresa puede verse comprometida. La ausencia de técnicas de seguridad estándar es un obstáculo para la adopción de la tecnología.

La calidad de un Web Services es un parámetro que no queda demasiado claro, pero cuya medida es fundamental a la hora de desarrollar un servicio maduro.

Esta tecnología está en desarrollo y la mayoría de los protocolos en los que se basa, aún no son estándar.

1.2 WEB SERVICES ACTUALIZADOR DE DATOS.

Conocida las características básicas del modelo de los Web Services o Servicios Web, enmarcaremos estos conocimientos, en el problema puntual a tratar.

Este problema consiste en la actualización de los datos personales del usuario y su posterior solicitud y envío a la o las empresas a las cuales el usuario quiere avisar de sus nuevos datos.

Un ejemplo típico de estos, es el cambio de domicilio, cuando una persona natural o empresa, cambia su domicilio esto afecta su relación con otro conjunto de empresas, las cuales deben ser notificadas de este cambio. Para realizar esto en

forma automática y simple, se pensó en integrar esto en un solo portal, donde el usuario determinará a qué empresas avisar de la actualización de sus datos, estos datos serán certificados por una empresa específica y luego las empresas estipuladas por el usuario tomarán estos paquetes y actualizarán sus propias BD.

A continuación se presenta el diagrama de flujo que describe los procesos básicos asociados.

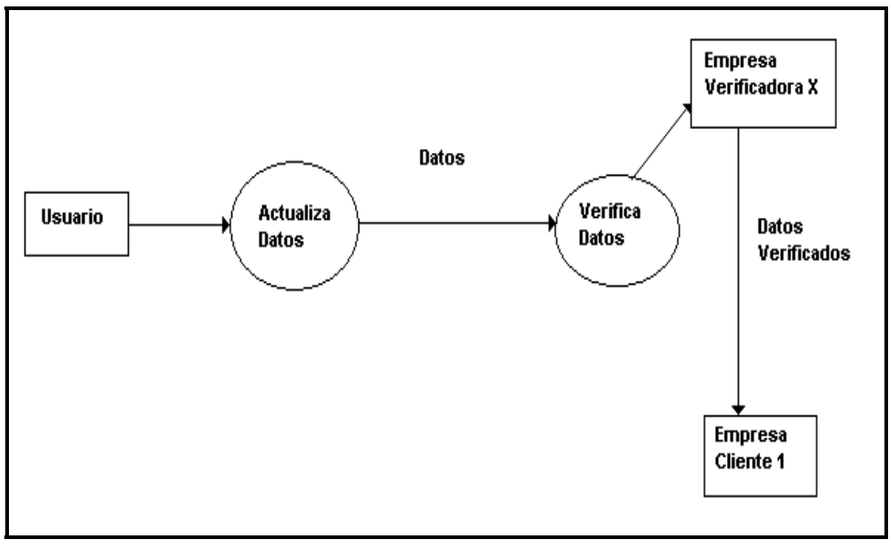


Figura 3, Diagrama de flujo

Como se observa en la Figura 3, el proceso básico es la transferencia de información entre las diferentes partes asociadas. El hecho de utilizar la tecnología de Web Services, es que debido a las características que presenta, hace que el manejo de información sea rápido y sin importar la plataforma en que trabajen las partes involucradas.

En los próximos capítulos se detallará con mayor precisión el manejo de la información en los Web Services.

1.3 OBJETIVOS

1.3.1 OBJETIVOS GENERALES.

- Formular e implementar un sistema de Servicios Web, enmarcados en un solo portal Internet, que permita entregar a determinadas empresas una reducción en los costos de administración de determinados servicios, brindándolos de una manera automática y más accesible, tanto para sus potenciales clientes como para ellas mismas.
- Implementar un prototipo escalable, en cuanto a la cantidad de módulos de software que pueden proveer servicios, para que los participantes del e-business, puedan optar por un mercado más amplio sin invertir demasiados recursos.

1.3.2 OBJETIVOS ESPECÍFICOS.

I. Investigar el estado actual del desarrollo de Web Services e identificar, aplicar metodologías y técnicas líderes actuales utilizadas por este tipo de servicios y de otros sitios de consumo masivo en Internet, desde el punto de vista comercial y tecnológico. A su vez investigar los distintos modelos de programación para Web Services que permiten dar solución a la problemática planteada.

II. Diseñar un modelo capaz de brindar ventajas cualitativas, en cuanto a la utilización de recursos, a las empresas que lo utilicen, otorgándoles información segura en sus transacciones comerciales con las empresas proveedoras.

III. Implementar un modelo de negocios y tecnológico que integre diferentes conceptos y que sea capaz de otorgar un servicio, asegurando la calidad de respuesta de éste, independiente de la plataforma tecnológica que opere en las empresas asociadas.

IV. Aplicación de tecnología de la familia W3C para las transacciones comerciales que se realicen, con el objeto de que estas sean transparentes para el usuario, en cuanto a tiempo de respuesta, sin importar el número de servicios que brinde el portal.

V. Definir el modelo y funcionalidades a implementar sobre el prototipo. Esto desde el punto de vista del Cliente, Servidor, Plataforma de desarrollo e Integración de Herramientas.

VI. Mejorar el prototipo implementado, en base a pruebas a realizar.

CAPÍTULO 2

2. EVOLUCION DEL DESARROLLO DE INTERNET.

En sus inicios el desarrollo de aplicaciones sobre Internet se limitaba a difundir documentos estáticos, existían servidores Web que proveían, vía el protocolo de comunicación HTTP[3], información en formato HTML, que luego era desplegada por los browser. Por lo cual Internet se convirtió en un enorme repositorio de información plana que solo servia para ser consultada, es por ésto que las tecnologías mayormente utilizadas durante este período fueron solo HTTP y HTML.

En el modelo de aplicación web "tradicional" encontramos una importante limitación: la interacción comienza y termina en dos puntos claramente definidos: la petición del usuario y la respuesta de la aplicación informática. Únicamente son dos los "interlocutores" que participan en este proceso

En cada intercambio de información que se produce, la aplicación informática debe "construir" una página resultado en formato HTML para presentar la información al usuario.

En la siguiente figura se observa el escenario típico de funcionamiento de los inicios del desarrollo de aplicaciones sobre Internet. [Amo00]

---

[3] Hiper Text Tranfer Protocol

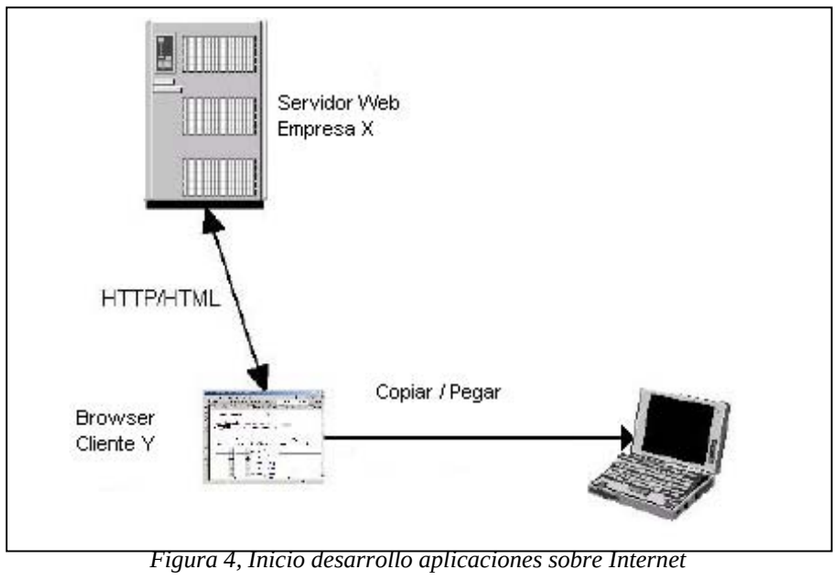


Figura 4, Inicio desarrollo aplicaciones sobre Internet

2.1 EVOLUCIÓN DE LAS APLICACIONES SOBRE INTERNET

A la par, las aplicaciones construidas bajo el modelo de 3 capas, capa de datos, lógica y de presentación, conocidas como aplicaciones cliente / servidor, se estaba enfrentando a una serie de limitantes. Entre las cuales se encontraban la poca modularidad de estas aplicaciones, la limitante que suponía un cambio de tecnología en la empresa que utilizaba dichas aplicaciones y por sobre todo su poca reusabilidad.[Eva98]

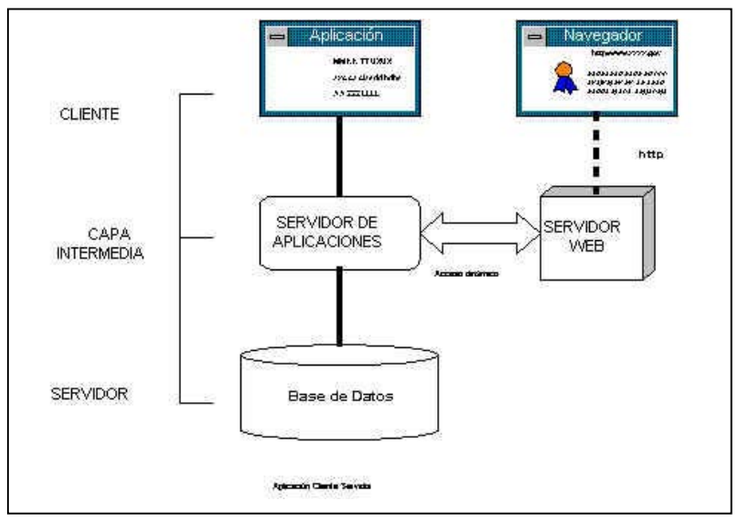


Figura 5, Modelo tres capas

En la siguiente figura se observa gráficamente la evolución de las aplicaciones sobre Internet:

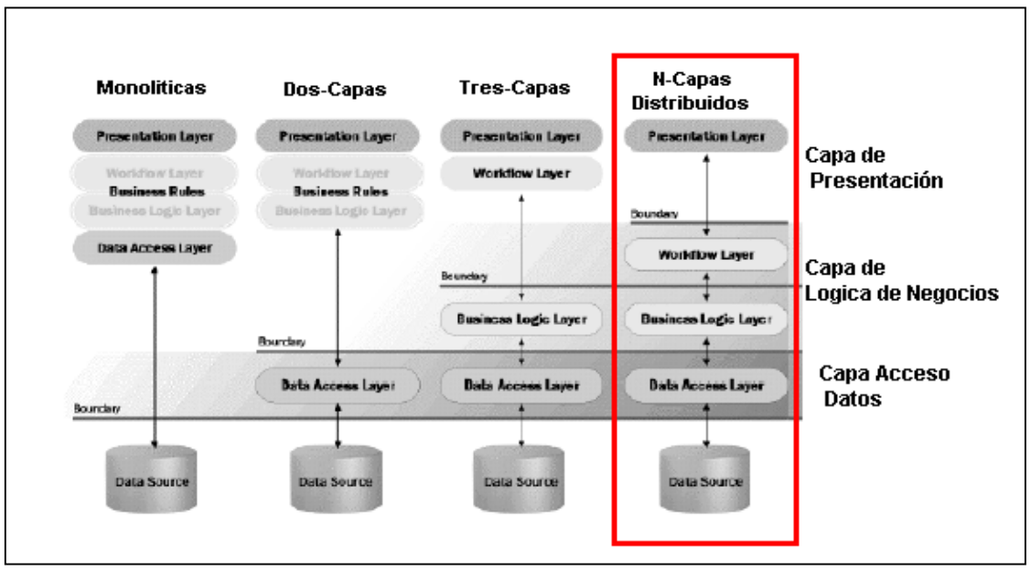


Figura 6, Evolución Aplicaciones

Desde la arquitectura conocida como cliente servidor, las aplicaciones han evolucionado, en busca de flexibilidad, escalabilidad y modularidad que no eran características de este tipo de arquitecturas.

Dadas estas problemáticas y al no encontrar respuestas claras a preguntas como ¿Qué sucedería si la aplicación informática tuviese que recurrir a un tercer sistema informático para satisfacer la petición cursada por el usuario?, es que nace la necesidad de ampliar este modelo, para permitir, la interacción de aplicaciones con módulos de otras aplicaciones. La respuesta a esto es lo que se conoce como modelo n-capas.

2.2 MODELO N-CAPAS.

El modelo n-tier o n-capas de informática distribuida ha emergido como la arquitectura predominante para la construcción de aplicaciones multiplataforma en la mayor parte las empresas de hoy. Este cambio radical en los modelos de desarrollo de aplicaciones, desde los sistemas monolíticos basados en mainframe y los tradicionales sistemas clientes-servidor, hacia sistemas distribuidos multiplataforma altamente modulables, representa simplemente la punta del iceberg de lo que ha llegado al mundo del desarrollo de aplicaciones.

Las arquitecturas basadas en n-capas son el siguiente paso lógico en un proceso de evolución, el cual, está basado en las arquitecturas convencionales cliente-servidor (2 y 3 capas) más la convergencia de dos tecnologías potentes:

1. Desarrollo de aplicaciones basadas en componentes, relacionado directamente con la programación orientada a objetos.
2. Internet, que representa un ejemplo de un sistema complejo de n capas cliente –servidor.

Los sistemas n-capas utilizan técnicas de desarrollo basadas en componentes combinados con los estándares abiertos de Internet, para crear aplicaciones multiplataforma muy potentes con bajos costos, facilidad de mantener y con gran efectividad. Lo que realmente es un nuevo modelo de n-capas es la posibilidad de distribuir objetos independientes sobre el número de capas que sean necesarias y enlazarlas dinámicamente, cuando sea necesario, para proporcionar una flexibilidad ilimitada a la aplicación.

Un punto importante a considerar del modelo de n-capas es que permiten trabajar con clientes distribuidos, tal como navegadores de Internet, PDAs, etc.

De este modo, las arquitecturas de n-capas se están posicionando rápidamente como la piedra angular de los desarrollos de aplicaciones empresariales y las compañías están adoptando esta estrategia a gran velocidad como mecanismo de posicionamiento en la nueva economía emergente que tiene su base en la red.

Actualmente la red es el computador, o el computador es la red. Este paradigma está creando un cambio fundamental en los modelos de computación que, a su vez, proporciona desafíos y oportunidades como nunca antes se había producido.

Como se observa, los componentes distribuidos de una arquitectura de n-capas son una tecnología esencial para crear la siguiente generación de aplicaciones e-business, aplicaciones que son altamente escalables, fiables y proporcionan un alto rendimiento e integración.

Las arquitecturas empresariales de n-capas se están convirtiendo en la nueva base para el desarrollo de aplicaciones de misión crítica y ofrecen la única arquitectura funcional para la siguiente generación de soluciones informáticas distribuidas basadas en Internet. Los sistemas distribuidos de n-capas proporcionan un conjunto de avances tecnológicos sin precedentes, como pooling de conexiones, multiplexado de conexiones, balanceo de carga, etc. Las aplicaciones industriales basadas en n-capas pueden ayudar a las compañías a desarrollar un nuevo núcleo de habilidades en prácticamente todo, desde la gestión del conocimiento hasta los sistemas relacionados con el comercio electrónico.

2.3 ARQUITECTURA MODELO N-CAPAS

2.3.1 VENTAJAS APPS. N-CAPAS CAPAS

Dentro de los beneficios de las aplicaciones de n-capas se encuentran:

- **Escalabilidad**. Habilidad de adicionar recursos para soportar mayores números de usuarios sin modificar las aplicaciones.
- **Extensibilidad**. Habilidad de incrementar la funcionalidad de una aplicación sin alterar lo ya existente.
- **Seguridad**. Capacidad de extender y propagar la autenticación en el "front-end" hacia las capas intermedias, salvaguardando la información de niveles de datos.
- **Administrabilidad**. Capacidad de realizar cambios en la lógica de negocios de manera sencilla y transparente para el usuario

2.3.2 DESVENTAJAS APPS. N-CAPAS CAPAS

Dentro de los puntos a considerar en el desarrollo de estas aplicaciones tenemos:

- Complejidad. En la medida en que existan mayores elementos en HW y SW, en momentos de falla la complejidad de detectarla es elevada.
- Comunicaciones. En la mayoría de los casos, cada capa se encuentra distribuida en la red, lo cual genera mayor consumo de ancho de banda.
- Costos de Mantenimiento. En la medida en que cada capa crece, tanto los costos de instalación, actualización y mantenimiento de HW y SW se Incrementan.

2.4 TECNOLOGÍAS MAS UTILIZADAS EN EL MODELO N CAPAS.

2.4.1 TECNOLOGÍAS DEL LADO DEL CLIENTE (HTML, XHTML)

Están insertadas en la página HTML del cliente y son interpretadas y ejecutadas por el navegador.

Es decir, que su correcta funcionalidad depende del soporte de la versión del browser a ser utilizado por el usuario visitante. [URL 4]

2.4.2 TECNOLOGÍAS DEL LADO DEL SERVIDOR (ASP, CGI, SERVLET)

Pueden o no estar insertadas dentro de la página HTML. (ASP, y PHP -que veremos más adelante- están embedidas en el código HTML). A diferencia del tipo anterior, estas tecnologías no dependen del navegador ya que son interpretadas y ejecutadas por el servidor. Por ejemplo, si utilizamos PHP en nuestro sitio necesitamos que el servidor donde esté alojado el mismo, tenga instalado PHP.

- **Servidor: ASP-Active-Server-Pages**

  Es una tecnología que deriva de VBScript (que a diferencia de ASP, éste es interpretado por el navegador) desarrollada por Microsoft. ASP corre en servidores Windows NT, y es una excelente opción para la creación de páginas dinámicas. Para ello, ASP se relaciona muy bien con bases de datos como ser SQL Server, entre otras.

- **CGI.**

  Common Gateway Interface fue la primera técnica utilizada para que el contenido de las páginas web se genere de manera dinámica, es común encontrar en los diferentes servidores web el módulo que soporta la ejecución de CGIs. ¿Pero en qué consiste exactamente este método de desarrollo web? De manera resumida se puede decir que el CGI es un mecanismo de comunicación entre el servidor WEB y una aplicación externa, esta aplicación puede estar desarrollada en casi cualquier lenguaje, esté sólo debe cumplir la condición de ser soportado por el servidor http, es común encontrar que la mayoría de las aplicaciones CGIs se encuentren desarrolladas con el lenguaje PERL.

- **Servlets**

  Son componentes del servidor. Estos componentes pueden ser ejecutados en cualquier plataforma o en cualquier servidor debido a la tecnología Java que se usa para implementarlos. Los Servlets incrementan la funcionalidad de una aplicación web. Se cargan de forma dinámica por el entorno de ejecución Java del servidor cuando se necesitan. Cuando se recibe una petición del cliente, el contenedor / servidor web inicia el servlet requerido. El Servlet procesa la petición del cliente y envía la respuesta de vuelta al contenedor / servidor, que es enrutada al cliente

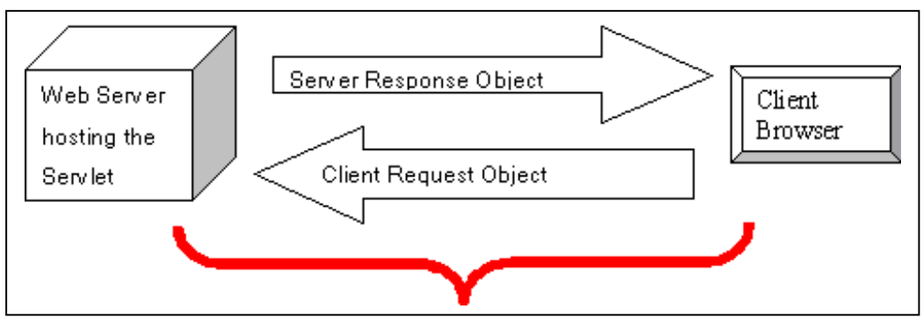


Figura 7, Modelo de respuesta a peticiones HTTP

La interacción cliente / servidor basada en Web usa el protocolo HTTP. EL protocolo HTTP es un protocolo sin estados basado en un modelo de petición y respuesta con un número pequeño y finito de métodos de petición como GET, POST, HEAD, OPTIONS, PUT, TRACE, DELETE, CONNECT, etc. La respuesta contiene el estado de la respuesta y meta-información describiendo dicha respuesta. La mayor parte de las aplicaciones web basadas en servlets se construyen en el marco de trabajo del modelo petición / respuesta HTTP.

2.5 EVOLUCIÓN DE DESARROLLO DESDE DE N-CAPAS A WEB SERVICES

En la actualidad, las aplicaciones de N niveles se han convertido en la pauta a seguir a la hora de generar software para empresas. Para casi todo el mundo, una aplicación de N niveles es aquella que se encuentra dividida en partes lógicas bien diferenciadas. Lo más común es que la elección de una división en tres partes (presentación, lógica empresarial y datos), aunque también existen otras posibilidades. A pesar de que las aplicaciones de N niveles surgieron en un principio para solucionar algunos de los problemas relacionados con las aplicaciones cliente / servidor tradicionales, con la llegada del Web esta arquitectura ha pasado a desempeñar un papel predominante en la nueva programación.

Como se verá más adelante, la arquitectura en la que se basan los Web Services, es una arquitectura distribuida, esta cualidad es una de las mayores asociaciones lógicas con el modelo de aplicaciones de n-capas. Esto se debe a que gran parte de su desarrollo se encuentra en diferentes plataformas y pueden estar en diferentes ubicaciones.

Además, si bien se siguen respetando las tres capas ya conocidas, en este tipo de desarrollo, puede existir más de una capa de presentación y la comunicación que existe entre la capa de negocios y la capa de presentación puede llegar a tener diferentes representantes.

En el siguiente capítulo se describe el modelo utilizado por los Web Services, así como se presentarán las características que hacen de él la nueva era en el desarrollo de aplicación web asociadas al modelo de n-capas.

CAPÍTULO 3

3. INTRODUCCIÓN A WEB SERVICES

En el presente capítulo, se profundiza los conceptos en referencia a los Web Services, ahondando en todas sus características y beneficios desde el punto de vista de arquitectura de software. [Gov01]

La programación basada en componentes se ha convertido en la más popular. Es difícil que en la actualidad se cree una aplicación que no involucre de alguna forma aprovechar componentes, normalmente de diferentes desarrolladores. Al ir creciendo en sofisticación las aplicaciones también ha crecido la necesidad de aprovechar componentes distribuidos en máquinas remotas.

Un ejemplo de aplicación basada en componentes es una solución de e-commerce entre extremos. Una aplicación residente en un sitio web necesitará enviar órdenes a una aplicación de Planificación de Recursos empresariales (ERP[4]) en el back-end. En la mayoría de los casos la aplicación ERP reside en un hardware diferente e incluso podría ejecutarse sobre un sistema operativo diferente. [Cab02]

El modelo de objetos componentes distribuidos de Microsoft (DCOM[5]), es una infraestructura de objetos distribuidos que permite a una aplicación invocar

---

[4] Enterprise Resource Plannig

[5] Microsoft Distriduted Component Object Model

componentes del modelo de objetos componentes (COM[6]) instalados en otro servidor, y se han portado a algunas plataformas distintas de Windows. Pero DCOM nunca ha conseguido una amplia aceptación en dichas plataformas, por lo que rara vez se utiliza para facilitar la comunicación entre computadoras con y sin Windows. Los fabricantes de software como ERP, suelen crear componentes para plataformas Windows, que se comunican con el sistema back-end, mediante protocolos propietarios.

Algunos servicios que se aprovechan para una aplicación de e-commerce, pueden no llegar a residir en el centro de procesamiento de datos. Por ejemplo, si la aplicación de e-commerce acepta pago mediante tarjeta de crédito, de las compras realizadas por un cliente, es necesario acceder a los servicios del banco asociado, para procesar la información de la tarjeta de crédito del cliente. Pero para efectos prácticos, DCOM y las tecnologías asociadas como CORBA y java RMI están limitadas a aplicaciones y componentes instalados en el mismo centro de datos corporativos. Las dos principales razones para ello son que de manera predeterminada estas tecnologías utilizan protocolos propietarios y dichos protocolos son inherentemente orientados a conexión.

La comunicación de los clientes con el servidor a través de Internet afronta numerosas barreras para la comunicación con el servidor. Los administradores, de cualquier parte de mundo, conscientes de la seguridad han implantado router y servidores de seguridad corporativos (firewalls) que deshabilitan prácticamente cualquier tipo de comunicación por Internet. Se puede considerar un “acto divino” conseguir que un administrador de red habilite puertos más allá del mínimo básico.

---

[6] Component Object model

Si se logra conseguir que el administrador de la red abra los puertos apropiados para los servicios, puede que los clientes no sean afortunados**.** El resultado es que los protocolos propietarios como los utilizados por DCOM, CORBA y Java RMI, no resultan prácticos en un escenario como Internet.

El otro problema con estas tecnologías, como ya se ha mencionado, es que son inherentemente orientadas a la conexión y, por tanto, no pueden manejar de forma apropiada interrupciones en la red. Si se produce una interrupción en la red, la próxima llamada que haga el cliente al servidor puede fallar.

La naturaleza orientada a conexión de estas tecnologías también convierte en un reto construir infraestructuras de equilibrio de carga, necesarias para proporcionar una alta escalabilidad. Cuando una conexión entre el cliente y el servidor se sobrecarga, no se puede enrutar la siguiente solicitud a otro servidor.

Los desarrolladores han intentado superar estas limitaciones aprovechando un modelo llamado programación sin estado, pero con un éxito limitado, ya que estas tecnologías son relativamente pesadas y convierte en caro el restablecimiento de una conexión con un cliente remoto.

Como el procesamiento de la tarjeta de crédito de un cliente se realiza en un servidor remoto en Internet, DCOM no es ideal para facilitar la comunicación entre el cliente e-commerce y el servidor que procesa la tarjeta de crédito. Como en una solución de ERP, normalmente se instala un componente de terceros dentro del centro de datos del cliente, en este caso por el proveedor de la solución de procesamiento de la tarjeta de crédito. Este componente sirve de poco más que de proxy que facilita la comunicación entre el software de e-commerce y el banco correspondiente utilizando un protocolo propietario.

3.1 ¿POR QUE WEB SERVICES?

Microsoft adoptó inicialmente, para soportar mejor estos escenarios de Internet, la estrategia de aumentar sus tecnologías existentes, incluyendo servicios Internet de COM(CIS7), que permite establecer una conexión DCOM entre el cliente y el componente remoto utilizando el puerto 80 o HTTP, pero existieron múltiples razones por las que CIS no fué aceptado.

De acuerdo a lo anterior, se necesitó una nueva solución que permitiera enfrentar tanto a los desarrolladores como a los usuarios, de mejor forma, los desafíos que presentaban las aplicaciones sobre Internet.

Por esto nacen los “Web Services”, que expone una interfaz para invocar una determinada actividad como intermediario del cliente. Un cliente puede acceder a un servicio web utilizando los estándares de Internet.

Dentro de los requisitos que cumple esta solución están:

- **Interoperabilidad**: un servicio remoto debe permitir su utilización por clientes de otras plataformas.
- **Amigabilidad con Internet**: La solución debe poder funcionar para soportar clientes que accedan a los servicios remotos desde Internet, y por lo tanto asumir todas las restricciones y funcionamiento de los servicios sobre Internet.
- **Interfaces de datos estándares:** No debería haber ambigüedad acerca del tipo de dato enviado y recibido desde un servicio remoto. Más aún, los tipos de datos definidos en el servicio remoto deben poderse corresponder

---

[7] COM Internet Services

razonablemente con los tipos de datos de la mayoría de los lenguajes de programación procedimentales.

- **Posibilidad de aprovechar los estándares de Internet existentes:** La implementación del servicio remoto debería aprovechar estándares de Internet existentes tanto como sea posible y evitar reinventar soluciones a problemas que ya se han resuelto. Una solución construida sobre un estándar de Internet ampliamente adoptado puede aprovechar conjuntos de herramientas y productos existentes creados para dicha tecnología.
- **Soporte para cualquier lenguaje:** no está ligada a un lenguaje de programación particular. Java RMI, por ejemplo, está ligada completamente al lenguaje Java. Sería muy difícil invocar funcionalidad de un objeto Java remoto desde Visual Basic o Perl. Un cliente debería ser capaz de implementar un nuevo servicio Web o utilizar un servicio Web existente, independientemente del lenguaje de programación en el que se haya escrito el cliente
- **Soporte para cualquier infraestructura de componentes distribuida:** no está fuertemente ligada a una infraestructura de componentes en particular. De hecho, no se debería requerir el comprar, instalar o mantener una infraestructura de objetos distribuidos, solo construir un nuevo servicio remoto o utilizar un servicio existente. Los protocolos subyacentes deberían proporcionar un nivel base de comunicación entre infraestructuras de objetos distribuidos existentes tales como DCOM y CORBA.[Tab02]

3.2 BLOQUES CONSTRUCTIVOS DE SERVICIOS WEB

En la siguiente figura se muestran los bloques constructivos principales necesarios para facilitar las comunicaciones remotas entre dos aplicaciones.

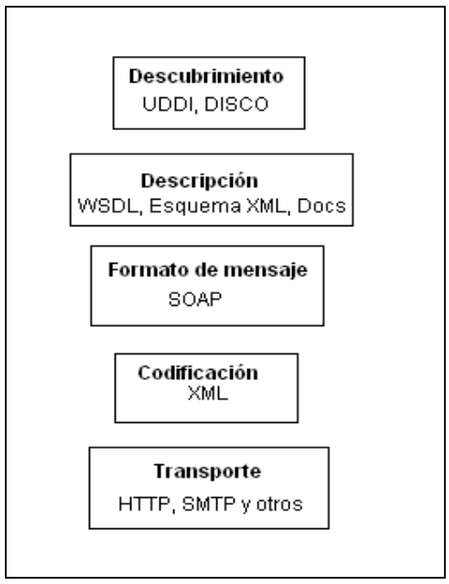


Figura 8, Bloques Constructivos servicios Web

- **Descubrimiento*:*** La aplicación cliente que necesita acceder a la funcionalidad que expone un Servicio Web necesita una forma de resolver la ubicación del servicio remoto. Se logra mediante un proceso llamado normalmente "Descubrimiento". El descubrimiento se puede proporcionar mediante un directorio centralizado así como por otros métodos. En DCOM, el servicio de descubrimiento lo proporciona el Administrador de control de servicios (SCM[8])

- **Descripción:** una vez que se ha resuelto el extremo de un servicio Web, el cliente necesita suficiente información para interactuar adecuadamente con el

[8] Service Control Manager

mismo. La descripción de un servicio Web implica metadatos estructurados sobre la interfaz que intenta utilizar la aplicación cliente así como documentación escrita sobre el servicio web incluyendo ejemplos de uso. Un componente DCOM expone metadatos estructurados sobre sus interfaces mediante una librería de tipos. Los metadatos dentro de una librería de tipos de componente, se guardan en un formato binario propietario a los que se accede mediante una Interfaz de programación de aplicaciones propietaria (API).

- **Formato del mensaje**: Para el intercambio de datos, el cliente y el servidor tienen que estar de acuerdo en un mecanismo común de codificación y formato de los mensajes.

  El uso de un mecanismo estándar de codificar los datos asegura que los datos que codifica el cliente los interpretara correctamente el servidor. En DCOM los mensajes que se envían entre un cliente y un servidor tienen un formato definido por el protocolo DCOM Object RPC (ORPC).

  Sin un mecanismo estándar para dar formato a los mensajes, desarrollar un conjunto de herramientas para abstraer al desarrollador de los protocolos subyacentes resulta casi imposible. Creando un nivel abstracción entre el desarrollador y los protocolos subyacentes permite al desarrollador centrarse más en el problema de negocio y menos en la infraestructura necesaria para implementar la solución.

  - **Codificación:** los datos que se transmiten entre el cliente y el servidor necesitan codificarse en el cuerpo del mensaje. DCOM utiliza un esquema

de codificación binaria para señalizar los datos de los parámetros que se intercambian entre el cliente y el servidor.

- **Transporte:** una vez que se ha dado formato al mensaje y se han serializado los datos dentro del cuerpo del mensaje, se debe transferir entre el cliente y el servidor utilizando algún protocolo de transporte. DCOM dispone de varios protocolos propietarios como TCP, NetBEUI, etc [URL 3].

3.3 ELECCIONES DE DISEÑO AL IMPLEMENTAR WEB SERVICES

A continuación se exponen algunas de las decisiones de diseño de los bloques constructivos, para los servicios Web. [Bri02]

3.3.1 ELECCIÓN DE PROTOCOLOS DE TRANSPORTE

Debido a que el cliente y el servidor podrían residir en la misma Red de Área Local, y además el cliente podría potencialmente comunicarse con el servidor por Internet, entonces el protocolo de transporte debería ser igual de utilizable en entornos de redes de área local y en Internet [Cur01].

Como se mencionó anteriormente, las tecnologías como DCOM, CORBA y Java RMI no resultan apropiados para la comunicación cliente-servidor por Internet. Protocolos como el HTTP[9] y el protocolo simple de transferencia de correo

[9] Hypertext Transfer Protocol

SMTP[10], son protocolos de Internet bien probados. HTTP, define un patrón de mensajes de petición / respuesta para enviar una petición y conseguir la respuesta asociada. SMTP, define un protocolo de mensajes enrutables para comunicación asíncrona.

Dentro de las ventajas de estos protocolos de comunicación están:

- Las aplicaciones Web, son inherentemente sin estado. No confían en una conexión permanente entre el cliente y el servidor. Es por esto que HTTP es un protocolo ideal, ya que si el servidor que sirve la petición original del cliente, queda fuera de servicio, las siguientes peticiones se pueden enrutar automáticamente a otro servidor siendo ésto transparente al cliente.
-
- SMTP, es ampliamente utilizado por la mayoría de las empresas, ya que apropiado para comunicaciones asíncronas. Si se interrumpe el servicio, la infraestructura del e-mail maneja automáticamente los reintentos. A diferencia de HTTP, se pueden enviar mensajes de SMTP a un servidor local de correo, que intentará enviar los mensajes en su lugar.

- Por último, tanto HTTP como SMTP, son de amplia utilización, la mayoría de las compañías confían en ellos para su utilización dentro de Internet.

[10] Simple Mail Transfer Protocol

3.3.2 ELECCIÓN DE UN ESQUEMA DE CODIFICACIÓN

HTTP y SMTP proporcionan un mecanismo para enviar datos entre el cliente y el servidor. Sin embargo, ninguno de ellos específica como se deben codificar los datos en el cuerpo del mensaje. [Gra92]

Con el objetivo de aprovechar los protocolos de Internet, la elección fue el lenguaje de marcado extensible XML[11].

XML ofrece muchas ventajas como son:

- Soporte multiplataforma
- Un sistema de tipos de datos común
- Soporte para un conjunto de caracteres estándar de la industria, evita el problema de codificación binaria.

RMI en cambio, no ha resuelto el problema de compatibilidad entre diferentes plataformas. Por ejemplo diferentes plataformas hardware tienen distintas representaciones binarias internas para los números de varios bytes. DCOM y CORBA, utilizan métodos binarios de codificación, que resultan problemáticos y necesitan una infraestructura de soporte para abstraer al desarrollador de los detalles.

---

[11] Extensible Markup Language

3.3.3 ELECCIÓN DE UN CONVENIO DE FORMATO

Suele ser necesario incluir metadatos adicionales en el cuerpo del mensaje, como información del tipo de servicios, en este caso XML no proporciona mecanismos para identificar entre el cuerpo del mensaje y sus datos asociados. Los protocolos de transporte como HTTP proporcionan un mecanismo extensible de cabeceras de datos, pero algunos datos asociados al menaje podrían no ser específicos del protocolo de transporte.

Es por esto que se decidió por la utilización del protocolo simple de acceso a objetos SOAP[12], que proporciona un mecanismo independiente del protocolo para asociar información de la cabecera con el resto del cuerpo del mensaje además de no imponer restricciones sobre el formato del mensaje.

3.3.4 ELECCIÓN DE LOS MECANISMOS DE DESCRIPCIÓN

SOAP proporciona una manera estándar de dar formato a los mensajes que se intercambian entre el servicio web y el cliente. Sin embargo el cliente necesita información adicional para serializar apropiadamente la solicitud e interpretar la respuesta.

El esquema XML proporciona un mecanismo para crear esquemas, que se puede utilizar para describir el contenido de un mensaje (también puede crear sus propios tipos de datos).

---

[12] Simple Object Access Protocol

Un esquema contiene un conjunto de tipos de datos y definiciones de elementos. Un servicio web utiliza el esquema no solo para comunicar el tipo de datos que se espera en un mensaje, sino también para validar los mensajes de entrada y salida. Sin embargo, un esquema por si solo no proporciona suficiente información para describir completamente un servicio Web. El esquema no describe los patrones de mensajes entre el cliente y el servidor.

Esta información se proporciona en un documento, en el lenguaje de descripción de Web Services WSDL[13] , que es un documento XML, que describe completamente un cierto servicio Web.

3.3.5 ELECCIÓN DE LOS MECANISMOS DE DESCUBRIMIENTO

Una vez desarrollado y documentado un servicio Web, debemos hacer que los clientes potenciales puedan localizarlo. Es por esto que se necesita una forma común de anunciar los servicios Web. UDDI[14] proporciona este mecanismo.

UDDI, es un directorio centralizado estándar de la industria, que se puede utilizar para anunciar y localizar servicios Web, que permite a los usuarios buscar servicios utilizando ciertos criterios de búsqueda, incluyendo el nombre de la compañía, categoría y tipo de servicio Web.

A continuación se describirá con más detalle cada uno de los protocolos y tecnologías mencionadas.

---

[13] Web Services Description Language

[14] Universal Description, Discovery, and Integration

3.3.6 SOAP

SOAP es un formato de mensaje que permite que dos sistemas de software se comuniquen independientemente de la plataforma de software y hardware que utilicen los dos equipos participantes.

Esta tarea se logra con el uso de estándares de la industria tales como HTTP y XML. La comunicación suele adoptar la forma de la solicitud de información y, a continuación, la respuesta a esa solicitud.

En la figura 9, un cliente A solicita un precio de almacén a otro cliente B mediante un mensaje SOAP que define el artículo que interesa al cliente A. El cliente B recibe la solicitud, interpreta el mensaje SOAP y busca la solicitud en su base de datos. El cliente B crea una respuesta de SOAP incluyendo el precio del artículo solicitado y la devuelve al cliente A. El cliente A recibe el mensaje de SOAP, lo interpreta y después lo representa en pantalla para el usuario.

SOAP facilita esta cooperación de datos estandarizando el formato y la estructura del mensaje que se pasa entre los equipos.

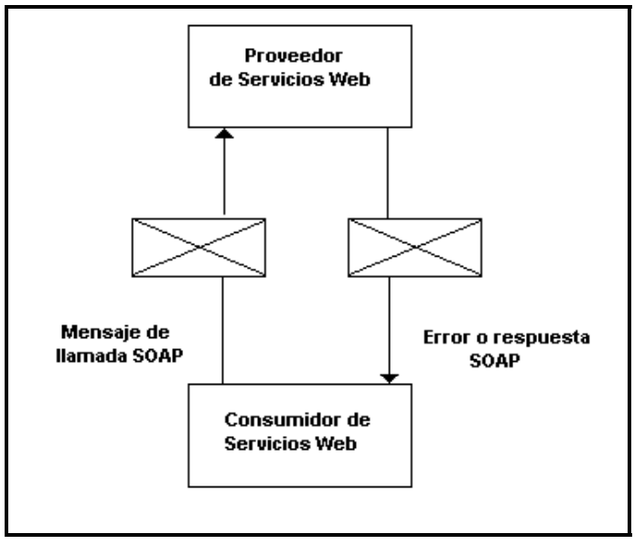


Figura 9, Solicitud / Respuesta de SOAP

3.3.6.1 Componentes de un mensaje SOAP

Los principales componentes de un mensaje SOAP son los siguientes:

- **Sobre de SOAP**: este contenedor para el encabezado y el cuerpo puede contener declaraciones de espacios de nombres XML, atributos y demás información (que debe acompañar al cuerpo del mensaje de SOAP)

- **Encabezado de SOAP**: esta sección contiene información opcional comprensible o no para el consumidor.

- **Cuerpo de SOAP**: esta sección contiene los datos reales de la llamada o de la respuesta del método (dependiendo del contexto de la solución que ofrece el Servicio Web).

A continuación se representa la composición de un mensaje de SOAP

Figura 10, Estructura de un mensaje SOAP

Otro punto importante de mencionar es el flujo de los mensajes SOAP. En la siguiente figura se podrá observar dicho flujo entre un servicio Web y una aplicación cliente que consume este servicio.

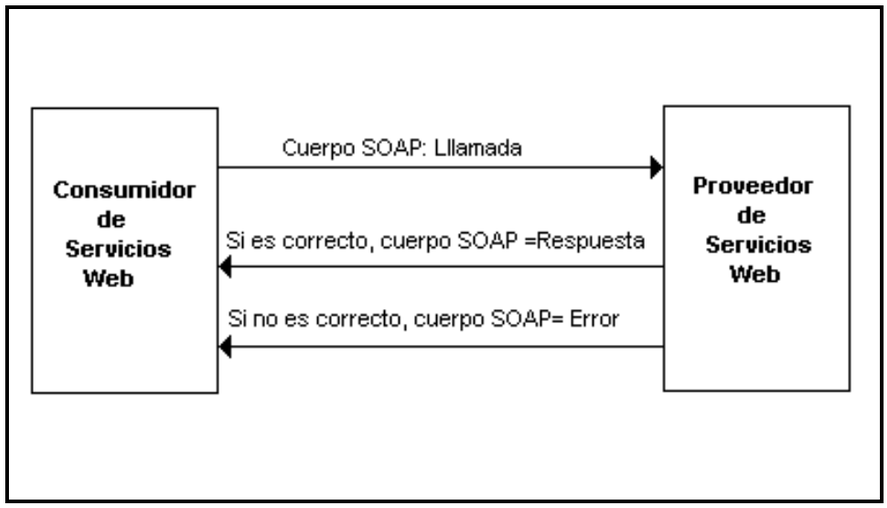


Figura 11, Flujo de los mensajes SOAP

3.3.7 WSDL

Como se ha podido ver SOAP, no integra ningún mecanismo que describa los servicios Web, y éstos no resultan descriptivos por sí mismos. Por esta razón es necesario un archivo adicional que describa los métodos, los parámetros y los tipos de datos del servicio Web, éste es el propósito de WSDL.[Chr01]

Aunque no es necesario, un documento de WSDL ofrece a los usuarios potenciales del servicio Web todo lo que necesitan saber para tener acceso al servicio Web, incluyendo la ubicación, la información de parámetros y la compatibilidad del protocolo. Si el entorno de desarrollo del usuario admite descubrimiento automatizado de servicios Web, el usuario puede actuar sobre el archivo WSDL de su servicio Web, y construir un proxy para simplificar la llamada al servicio. [URL 2]

3.3.7.1 Funcionamiento de WSDL

Cuando los clientes desean usar el servicio Web, se les dirige hacia el URI (ubicación) del documento WSDL manualmente o a través de cierto mecanismo automatizado (como UDDI o DISCO). Normalmente se crea un acceso al archivo WSDL y el código necesario para tener acceso al servicio Web, según la especificación WSDL. Este código se denomina proxy del servicio Web; y facilita la forma de interactuar con el servicio Web encapsulando la complejidad de construir la llamada apropiada y de enviarla a la red.

En la siguiente figura se observa con más detalle lo descrito anteriormente:

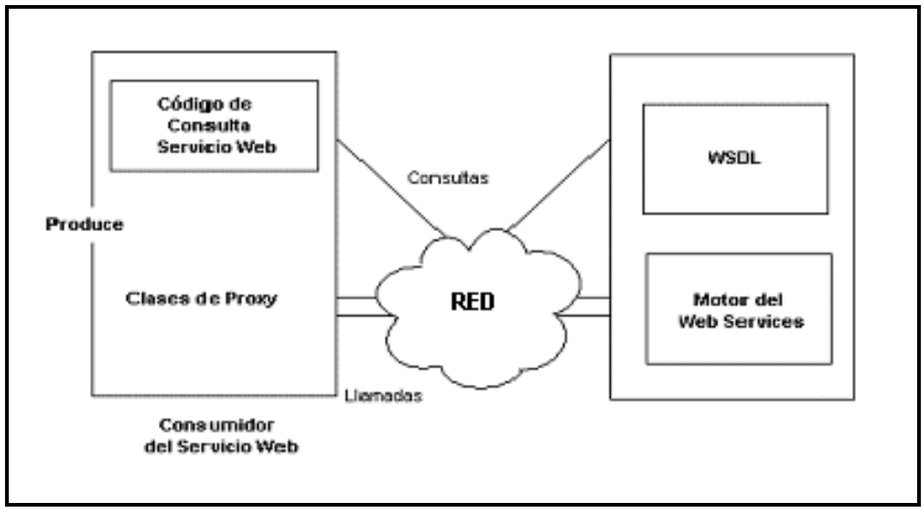


Figura 12, Uso de WSDL para crear proxy para el servicio Web

3.3.8 DISCO

DISCO[15] , es una especificación creada para ayudar a determinar como pueden encontrar los servicios Web en un servidor. Un archivo DISCO contiene el identificador de recursos uniforme (URI[16]), de cada servicio Web disponible en ese equipo. El URI apunta normalmente al documento WSDL, que a su vez apunta al servicio Web en cuestión.

3.3.9 UDDI

Se puede concebir como el motor de búsqueda para los servicios Web. Es una especificación que define como se puede crear un directorio de negocios que contiene una serie de referencias a las compañías que ofrecen servicios Web en línea; también guarda información adicional sobre la compañía que ofrece los servicios Web, como su clasificación de negocio, la ubicación geográfica, etc. Cuando se encuentra un negocio en el motor de búsqueda, se ve información para saber más sobre los procesos de negocio de la compañía y sus servicios Web. El directorio suele ofrecer un URI que representa un archivo DISCO o WSDL en el servidor de la compañía. [URL 6]

La idea que subyace a UDDI es que es un lugar de reunión para las compañías que buscan mercancías y servicios y aquellas que proveen las mercancías y servicios, y un lugar en donde ambas partes desean realizar el negocio electrónicamente.

---

[15] Discovery of Web Services

[16] Uniform Resource Identifier

La siguiente figura ilustra la relación entre los servicios Web y las tecnologías adicionales descritas en los puntos anteriores.

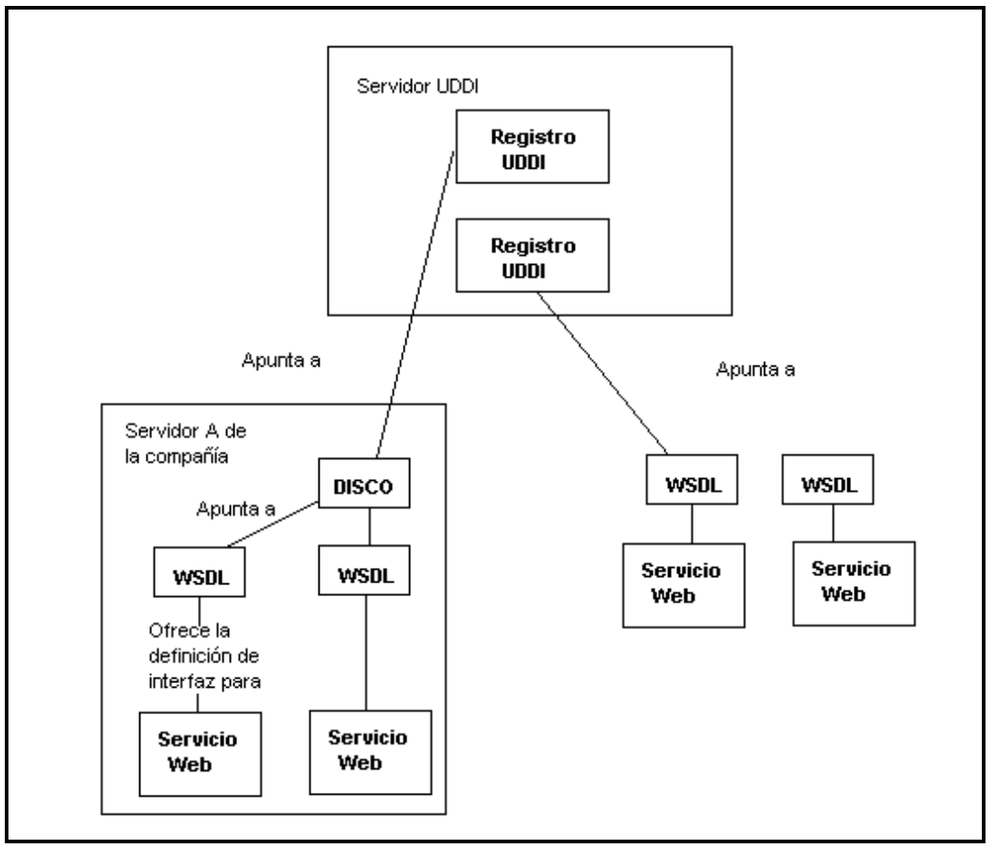


Figura 13, Relación entre UDDI, DISCO, WSDL y los servicios Web

Los servicios Web se pueden llamar y, a continuación, devolver los resultados mediante SOAP. Para que un socio potencial entienda lo que ofrece el servicio Web de una compañía, debe conocer la compañía, la ubicación de su servidor, el URI de sus archivos WSDL o DISCO, o encontrar la compañía en un motor de búsqueda de UDDI. Después de haber encontrado la compañía, el socio puede calcular sus intereses financieros y hacer que sus profesionales creen el código para consumir el servicio Web.

3.4 PRÓXIMO PASO EN LOS SERVICIOS WEB.

Dentro de las omisiones que se han hecho y que se mantienen en la infraestructura de componentes distribuidos están:

- **Falta de una API específica para servicios Web:** muchas infraestructuras de componentes distribuidos definen una API para realizar tareas como la iniciación del sistema de ejecución, la creación de instancias de componentes, etc. En el caso de los Servicios Web, al ser estos independientes del lenguaje de programación, no existe una API, asociada a un lenguaje determinado.

- **Servicio de componentes:** la plataforma de servicios Web no proporciona muchos de los servicios que se encuentran normalmente en infraestructuras de componentes distribuidos, como la administración del período de vida de objetos remotos, el soporte para transacciones distribuidas, etc. Estos servicios se dejan a la implementación de la infraestructura de componentes distribuidos.

Luego de esta visión general de la estructura y cualidades de los Web Services, en el siguiente capítulo se presenta el diseño y arquitectura del prototipo a crear, basándose en todos los aspectos revisados en la sección actual.

CAPÍTULO 4

4. DISEÑO Y ARQUITECTURA DE UN PROTOTIPO DE WEB SERVICES

Una de las áreas de desarrollo tecnológico que hoy en día han alcanzo mucha importancia para las empresas, es el manejo y transferencia de datos de clientes. Muchas veces para la actualización de datos personales de los clientes, estos deben realizar el proceso en las diferentes empresas de las que sean clientes, con lo cual repiten el proceso la cantidad de veces que sea necesario y en cada empresa distinta.

Esta sección del documento da a conocer el proceso de diseño de un Web Services, en particular se presenta un modelo de negocio para el de manejo y actualización de datos en línea, permitiendo así centralizar este trámite en un solo lugar para ser utilizado por empresas externas y automatizando dicho proceso con las ventajas que ofrece Internet y la utilización de Web Services.

Dentro de este proyecto se ha delimitado varias etapas del proceso completo de actualización que va desde el ingreso de datos hasta la entrega de los datos validados, en donde seria muy conveniente contar con servicios web automáticos que permitan agilizar el proceso.

4.1 PERSPECTIVA DE NEGOCIOS.

La temática particular que aborda este prototipo, corresponde a generar una aplicación o Servicio Web que permita a un cliente cualquiera, inscrito en el sistema actualizar datos personales, validar la información entrega y posteriormente hacer entrega de dicha información a un conjunto de empresas clientes, relacionadas al usuario inicial.

Una de las ventajas de usar el modelo implementado por los Web Services, que al ser modular, este puede ser instanciado por diferentes sistemas. En este caso particular, el Web Services de Actualización de Datos, fue pensado para ser llamado desde una aplicación Web, que se comunicara con él, vía mensajes SOAP XML.

El funcionamiento principal de este servicio Web será la obtención de los datos ingresado por el cliente y el posterior envió de estos a la entidad validadora por ejemplo DICOM, esta última, devolverá estos datos, con la respuesta de confirmación de su validación. Todo este envió de información, será como se mencionó con anterioridad, utilizando el protocolo SOAP, con mensajes en XML.

También se debe mencionar, que gracias al uso de una “pizarra electrónica”, el usuario dispondrá de una lista de empresas inscritas en el sistema, de donde podrá seleccionar a que empresa cliente desea proveer o actualizar su información personal.

El uso que se le dará en la aplicación Web, será que una empresa cliente podrá acceder a estos datos, previa inscripción en el sitio, luego podrá ingresar los datos a actualizar, dependiendo de una serie de opciones que dicha aplicación Web

podrá presentar. Una vez almacenados estos datos, serán enviados a la empresa encargada de validar que la información enviada sea correcta, al constatarse que así es, estos datos serán enviados a la empresa cliente que el usuario seleccionó. Una vez que la empresa cliente ha recibido la información completa, la aplicación Web notificara al cliente que el proceso se realizó a cabo satisfactoriamente.

Desde el punto de vista de negocios, este sistema representa una base para dar a conocer que los sistemas, pueden modularizarse y permitir que múltiples aplicaciones en diferentes plataformas, utilicen todas o parte de las funcionalidades que estos poseen. Esto permitirá una expansión en cuanto al tipo de aplicaciones que se desarrollarán en el futuro, enfocándose más en sistemas distribuidos, y reutilizables.

4.1.1 ALCANCES

- Análisis, diseño e implementación de cada uno de los módulos realizados.
- Contemplar todos los datos necesarios para la solicitud del servicio Web que se desee utilizar.
- Son servicios web integrados a una base de datos, sólo utilizados para el diseño general del sistema y que podrán ser utilizados por diferentes sistemas.

4.1.2 LIMITACIONES

Las limitantes que tendrá este proyecto, serán las siguientes

- Se enfocará mayoritariamente al diseño, construcción y consideraciones para la futura implementación del Servicio Web antes señalado.

- Como parte del proyecto se incluyen algunas pruebas prototipo de programas que muestran las funcionalidades básicas de las tecnologías estudiadas, la gran limitación es el tiempo para la programación y pruebas de los Servicios Web.

4.2 FUNCIONALIDADES DEL WEB SERVICES

Para describir con mayor detalle las funcionalidades que entregará nuestro servicio web, se presentan a continuación los diagramas de los ciclos de vida del sistema, tanto del lado cliente, como de la empresa cliente.

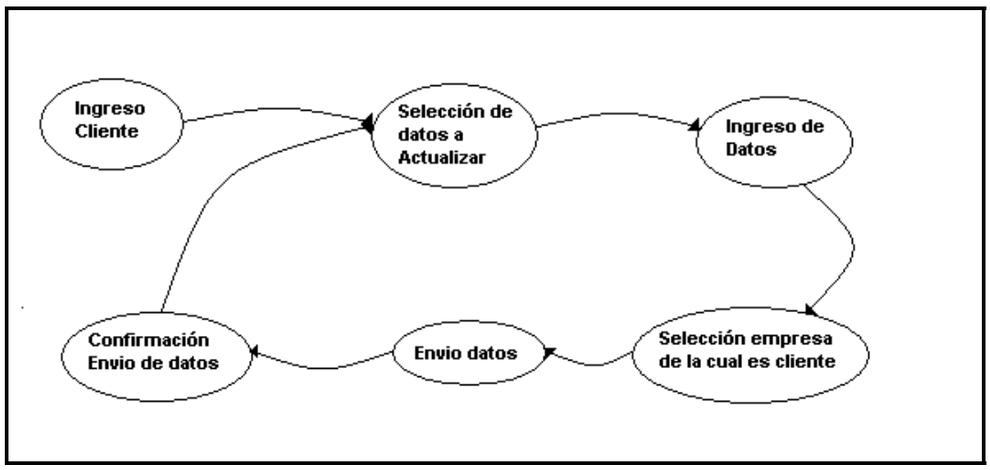


Figura 14, Ciclo de Vida Aplicación lado Cliente

Se ha considerado para el desarrollo de este sistema, que el Servicio Web, será utilizado por una aplicación Web.

Como se describe en la figura anterior, las funcionalidades básicas que estarán a disposición del usuario del sistema serán:

- **Ingreso al sistema:** previo registro en el sitio web, el usuario deberá ingresar su login y password, para acceder al sistema, esto provee un mayor nivel de seguridad y permite al usuario recuperar la información ingresada para su posterior actualización.
- **Ingreso de datos:** una vez seleccionado el tipo de dato a actualizar, se desplegará la información anteriormente registrada en el sitio, y el usuario tendrá la opción de modificarla. Si no existe diferencia con la anterior, se avisará al usuario, como un posible error.
- **Selección de empresa de la cual es cliente:** una vez actualizado el dato y grabado, el próximo paso será la selección de la empresa a la cual se le enviará la información del usuario actualizada. Por el momento y al ser este sólo un prototipo, se listará solo un conjunto pequeño de empresas.
- **Envío de datos:** completados todos los pasos anteriores, se procederá al envío de datos para su verificación, este paso es transparente para el usuario, y está a cargo de una empresa externa cuya función es validar la información actualizada por el usuario.
- **Confirmación envío de datos:** al usuario se le envía una confirmación de que sus datos fueron enviados satisfactoriamente en caso de no ocurrir problemas. En el caso de que la verificación de los datos entregados, reporte algún problema, se notificará al usuario vía mail de los problemas que existieron en el proceso.

Las funcionalidades descritas anteriormente son las que podrá utilizar un cliente normal inscrito con anterioridad en el sitio web. A continuación se presenta el ciclo de vida de la aplicación por el lado de la empresa cliente la cual recogerá los datos actualizados y verificados con anterioridad, ingresados por el cliente.

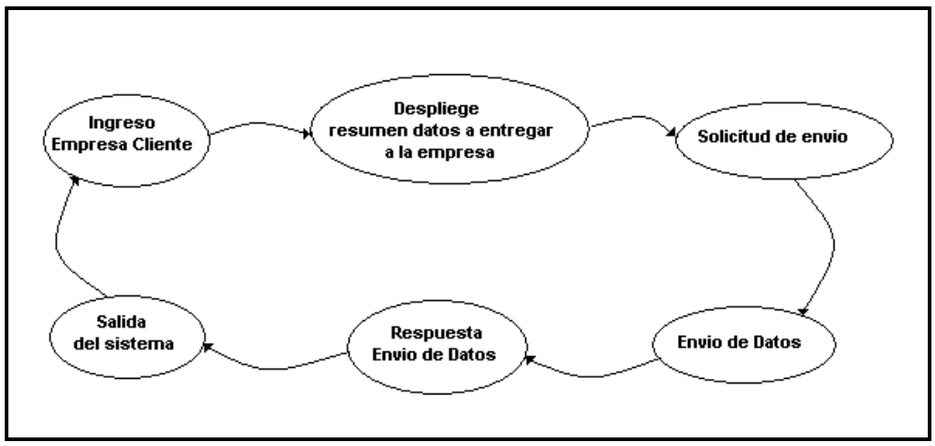


Figura 15, Ciclo de vida Aplicación, lado empresa cliente

Como se observa en la Figura 15, las funcionalidades que el sistema presentará para la empresa cliente serán las siguientes:

- **Ingreso Empresa cliente:** al igual que el cliente normal, el ingreso al sitio de la empresa cliente, dependerá de la previa inscripción en el sitio web. Luego de eso se le solicitará ingresar el login y password correspondiente.
- **Despliegue resumen datos a entregar a la empresa:** una vez que se ha ingresado al sitio web, el encargado, observará una serie de datos resumidos de los clientes que han actualizado sus datos en el sistema y solicitan que se actualicen en las respectivas bases de datos de la empresa. Esta última seleccionará los datos que actualizará, dependiendo del criterio que consideren adecuado.
- **Solicitud de envío:** Luego de la selección de los datos que la empresa recogerá, se solicita el envío de éstos.
- **Envío de datos:** cuando se recibe la confirmación del envío de datos, se comienza el traspaso de la información solicitada. Este paso es transparente para la empresa cliente.
- **Respuesta de envío de datos:** una vez que la transferencia haya sido realizada con exito, se notifica a la empresa cliente.

- **Salida del sistema:** cuando se ha efectuado el proceso completo se tiene la opción de abandonar el sistema, hasta la próxima entrega de datos.

4.3 UML Y UN MODELO DE DISEÑO DE NUESTRO WEB SERVICES.

El diseño corresponde al primer paso en la construcción de cualquier producto o sistema de información. Para el caso particular de este trabajo de titulación, el diseño de nuestro prototipo está presentado utilizando UML (Unified Modeling Language). Esta notación ha sido ampliamente aceptada debido al prestigio de sus creadores y debido a que incorpora las principales ventajas de cada uno de los métodos particulares en los que se basa: Booch, OMT y OOSE. UML ha puesto fin a las llamadas “guerras de métodos” que se han mantenido a lo largo de los 90, en las que los principales métodos sacaban nuevas versiones que incorporaban las técnicas de los demás. Con UML se fusiona la notación de estas técnicas para formar una herramienta compartida entre todos los ingenieros de software que trabajan en el desarrollo orientado a objetos.

4.3.1 INTRODUCCIÓN A UML.

La literatura define a UML como un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar objetos de un sistema de software. Se usa para entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir.

UML es también un lenguaje de modelamiento visual que permite una abstracción del sistema y sus componentes. Dentro de los objetivos fundamentales que persigue UML son; ser un lenguaje universal, como cualquier lenguaje de

propósito general e imponer un estándar mundial. UML debe entenderse como un estándar para modelado y no como un estándar de proceso de software. Aunque UML debe ser aplicado en el contexto de un proceso, la experiencia ha mostrado que organizaciones diferentes y dominios del problema diferentes requieren diferentes procesos. Por ello se han centrado los esfuerzos en un meta-modelo común (que unifica las semánticas) y una notación común que proporcione una representación de esas semánticas. De todas formas, los autores de UML fomentan un proceso guiado por casos de uso, centrado en la arquitectura, iterativo e incremental.

En particular UML mantiene una serie de herramientas o diagramas que permiten modelar un sistema dependiendo de su naturaleza. En este trabajo se utilizaron los diagramas de casos de usos, diagramas de secuencias y actividades, las cuales permiten de manera básica un modelamiento de sistemas web. Se presenta un recordatorio de los principales conceptos para poder comprender de mejor manera el trabajo presentado.

4.3.1.1 Casos de Uso y sus Elementos.

El diagrama de casos de uso representa la forma en como un cliente (Actor) opera con el sistema en desarrollo o a especificar, además de la forma, tipo y orden en como los elementos interactúan (operaciones o casos de uso).

A continuación se presentan los principales elementos utilizados en los casos de usos.

- **Actores**

Los actores son elementos que desempeñan un papel externo en el sistema que realiza algún tipo de interacción con el mismo, pueden ser personas u otros sistemas que suministran o envían información al sistema. Por otro lado se pueden distinguir los siguientes tipos de actores.

  - Principales: personas que usan o interactúan en el sistema.
  - Secundarios: personas que mantienen o administran el sistema.
  - Material externo: dispositivos materiales imprescindibles que forman parte del ámbito de la aplicación o sistema y deben ser utilizados.
  - Otros sistemas: sistemas con los que el sistema en modelamiento interactúa.

- **Caso de Uso**

El caso de uso es una función específica que se realiza tras una orden de algún agente externo, sea desde una petición de un actor o bien desde la invocación desde otro caso de uso. Dicho de otra forma, un caso de uso específica qué hará el sistema, pero no cómo.

Un caso de uso es una descripción de la secuencia de interacciones que se producen entre un actor y el sistema, cuando el actor usa el sistema para llevar a cabo una tarea específica.

- **Relaciones**

Es el tipo de relación más básica que indica la invocación desde un actor o caso de uso a otra función (caso de uso). Dicha relación se denota con una flecha simple.

- **Dependencia o Instanciación**

Es una forma muy particular de relación entre clases, en la cual una clase depende de otra, es decir, sé instancia (se crea). Dicha relación se denota con una flecha punteada.

- **Generalización**

Este tipo de relación es uno de los más utilizados, cumple una doble función dependiendo de su estereotipo, que puede ser de Uso (<<uses>>) o de Herencia (<<extends>>). Este tipo de relación esta orientado exclusivamente para casos de uso (y no para actores).

extends: Se recomienda utilizar cuando un caso de uso es similar a otro (características).

uses: Se recomienda utilizar cuando se tiene un conjunto de características que son similares en más de un caso de uso y no se desea mantener copiada la descripción de la característica.

- **Diagramas de Actividad**

El diagrama de actividad es un diagrama de flujo del proceso multi-propósito que se usa para modelar el comportamiento de un sistema. Los diagramas de actividad se pueden usar para modelar un Caso de Uso, o una clase, o un método complicado.

Un diagrama de actividad es parecido a un diagrama de flujo; la diferencia clave es que los diagramas de actividad pueden mostrar procesamiento en paralelo (parallel processing). Esto es importante cuando se usan diagramas de actividad para modelar procesos ’bussiness’, algunos de los cuales pueden actuar en paralelo, y para modelar varios hilos en los programas concurrentes. En general se utilizan para modelar los aspectos dinámicos de un sistema y describen los cambios de estado que experimenta un objeto.

- **Diagramas de Interacción**

En los diagramas de interacción se muestra un patrón de interacción entre objetos. Hay dos tipos de diagrama de interacción, ambos basados en la misma información, pero cada uno enfatizando un aspecto particular: Diagramas de Secuencia y Diagramas de Colaboración.

Un diagrama de Secuencia muestra una interacción ordenada según la secuencia temporal de eventos. En particular, muestra los objetos participantes en la interacción y los mensajes que intercambian ordenados según su secuencia en el tiempo.

El eje vertical representa el tiempo, y en el eje horizontal se colocan los objetos y actores participantes en la interacción, sin un orden prefijado. Cada objeto o actor

tiene una línea. El Diagrama de Secuencia es más adecuado para observar la perspectiva cronológica de las interacciones, muestra la secuencia explícita de mensajes y son mejores para especificaciones de tiempo real y para escenarios complejos.

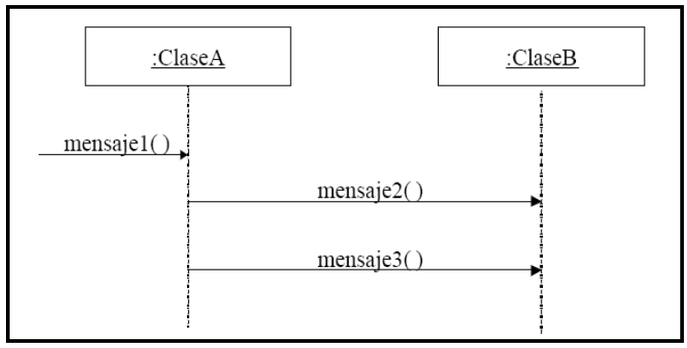


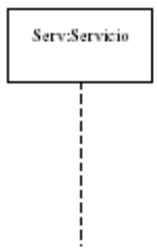
Figura 16, Ej. de Diagrama de Secuencia

Por otro lado un Diagrama de Colaboración muestra una interacción organizada basándose en los objetos que toman parte en la interacción y los enlaces entre los mismos (en cuanto a la interacción se refiere). A diferencia de los diagramas de secuencia, los diagramas de colaboración muestran las relaciones entre los roles de los objetos. La secuencia de los mensajes y los flujos de ejecución concurrentes deben determinarse explícitamente mediante números de secuencia. Ofrece una mejor visión espacial mostrando los enlaces de comunicación entre objetos, muestra las relaciones entre objetos y son mejores para comprender todos los efectos que tiene un objeto y para el diseño de procedimientos. El diagrama de colaboración puede obtenerse automáticamente a partir del correspondiente diagrama de secuencia.

A continuación se presentan los principales elementos utilizados para describir los objetos participantes en la interacción y los mensajes que intercambian ordenados según su secuencia en el tiempo.

- **Línea de vida de un objeto.**

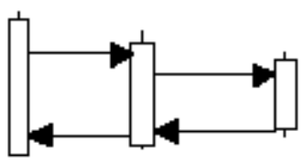
Un objeto se representa como una línea vertical punteada con un rectángulo de encabezado. El rectángulo de encabezado contiene el nombre del objeto y el de su clase, en un formato nombreObjeto: nombreClase.



Con rectángulos a través de la línea principal que denotan la ejecución de métodos.

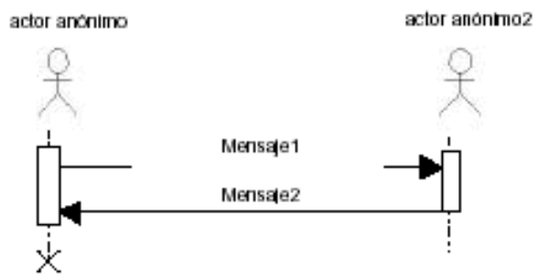
- **Activación**

Muestra el período de tiempo en el cual el objeto se encuentra desarrollando alguna operación, bien sea por sí mismo o por medio de delegación a alguno de sus atributos. Se denota como un rectángulo delgado sobre la línea de vida del objeto.



- **Mensaje**

El envío de mensajes entre objetos se denota mediante una línea sólida dirigida, desde el objeto que emite el mensaje hacia el objeto que lo ejecuta. Terminando este mensaje se da a la destrucción de la operación del objeto denotado por una "X" al finalizar la línea punteada.



4.4 CASOS DE USO

4.4.1 DEFINICIÓN DE ACTORES

Sistema usuario

Actor Principal: Administrador sistema usuario / usuario Final.

Rol: maneja envío datos a validar

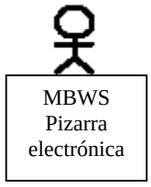
Empresa Validadora

Actor Secundario: Empresa Formal Validadora de Datos

Rol: Validar los datos entregados por el usuario.

Actor Principal: Empresa Cliente

Rol: Recupera los datos ya validados, entregados por el cliente



Actor Secundario: MBWS, pizarra electrónica

Rol: mostrar una lista de empresas asociadas al servicio

Actor Secundario: PWS (Proveedor del Web Services de Actualización de datos)

Rol: Ofrecer el Servicio de Actualización de Datos / coordinar la transferencia de toda la mensajería.

4.4.2 CASOS DE USO DEL WEB SERVICES

A continuación se observa el diagrama de casos de uso, para el diseño del Web Services de Actualización de datos.

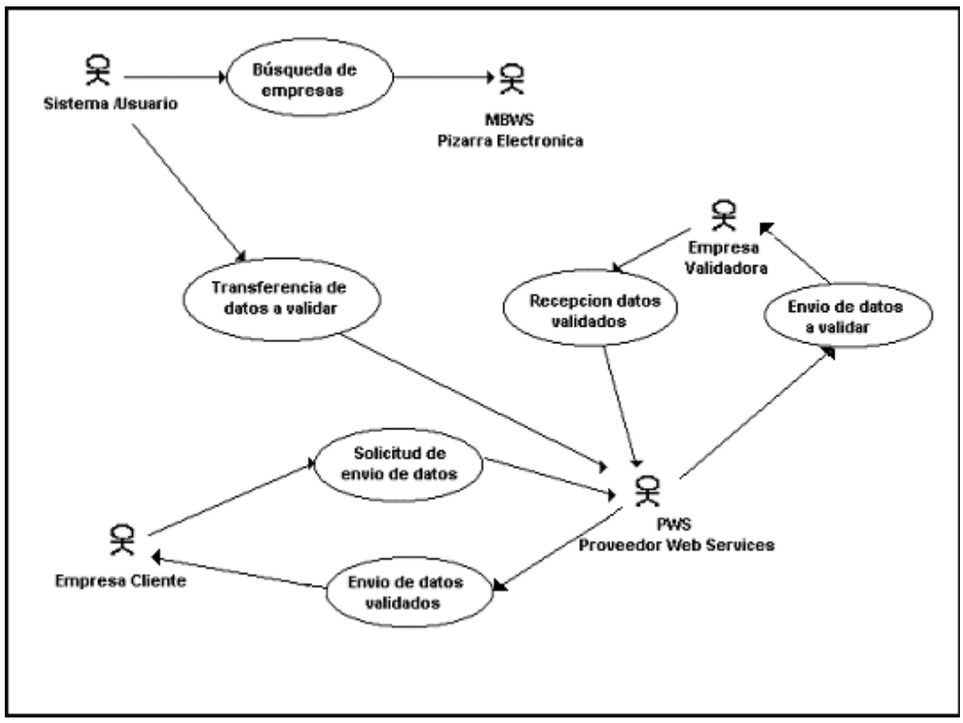


Figura 17, Diagrama casos de Usos

4.4.3 PLANTILLAS DE LOS CASOS DE USO

Nombre del Caso de Uso	**Búsqueda de empresas**
Actor Principal	Sistema /Usuario Final
Versión	1.0
Autor	Angela Reyes
Fecha	Marzo 2004
Actores involucrados	Sistema /Usuario Final: Obtener una lista de empresas inscritas. Pizarra electrónica (MBWS): Ofrecer la lista de varias empresas inscritas en el servicio.
Objetivos	1. Proporcionar una lista de empresas inscritas en el sistema, que actualizarán sus sistemas con la nueva información entregada por el usuario,
PreCondición	
Secuencia normal	El sistema cliente solicita una lista de empresas que cumplan con ciertas características MBWS proporciona la lista de empresas que cumplan con los requisitos solicitados
Excepciones	Que la lista de empresas quede vacía
Frecuencia esperada	Cada vez que el sistema cliente lo solicite

Comentarios	

Tabla 1. Planilla Caso de Uso: Búsqueda de empresas

Nombre del Caso de Uso	**Transferencia datos a validar**
Actor Principal	Sistema /Usuario Final PWS Proveedor Web Services
Versión	1.0
Autor	Angela Reyes
Fecha	Marzo 2004
Actores involucrados	Sistema /Usuario Final: Envía los datos a validar. PWS: Recibe y Distribuye los datos a la entidad validadora.
Objetivos	Validar la veracidad de la información del cliente.
PreCondición	
Secuencia normal	El sistema cliente envía la información que desea actualizar El PWS, gestiona la información recibida
Excepciones	
Frecuencia esperada	Cada vez que se tenga nueva información del cliente.
Comentarios	

Tabla 2. Planilla Caso de Uso: Transferencia datos a validar

Nombre del Caso de Uso	**Envío de datos a validar**
Actor Principal	PWS, proveedor Web Services Empresa Validadora
Versión	1.0
Autor	Angela Reyes
Fecha	Marzo 2004
Actores involucrados	PWS, proveedor Web Services: envía para su verificación, la información ingresada por el usuario Empresa Validadora: recibe los datos y verifica su validez
Objetivos	Validar información entrega da por el usuario
PreCondición	
Secuencia normal	El PWS, mantiene la información ingresada por el usuario y la envía a la empresa validadora La empresa Validadora, verifica que la información ingresada por el usuario sea correcta.
Excepciones	
Frecuencia esperada	Cada vez que se tenga nueva información del cliente.

Tabla 3. Planilla Caso de Uso: Envío de datos a validar

Nombre del Caso de Uso	**Recepción datos validados**
Actor Principal	PWS, proveedor Web Services Empresa Validadora
Versión	1.0
Autor	Angela Reyes
Fecha	Marzo 2004
Actores involucrados	PWS, proveedor Web Services: recibe la información validada Empresa Validadora: envía datos verificados
Objetivos	Recepción de datos verificados
PreCondición	
Secuencia normal	La empresa Validadora, envía la información validada El PWS, recibe la información verificada y los posibles comentarios asociados.
Excepciones	
Frecuencia esperada	Cada vez que se tenga nueva información del cliente.
Comentarios	

Tabla 4. Planilla Caso de Uso: Recepción datos validados

Nombre del Caso de Uso	**Solicitud de envío de datos**

Actor Principal	Empresa Cliente PWS, Proveedor Web Services
Versión	1.0
Autor	Angela Reyes
Fecha	Marzo 2004
Actores involucrados	Empresa Cliente: solicita el envío de datos a la empresa validadora PWS: recibe solicitud del envío de datos de una determinada empresa cliente
Objetivos	Solicitar datos validados
PreCondición	
Secuencia normal	La empresa cliente, solicita el envío de datos asociados a ella El PWS, recibe dicha solicitud e identifica la empresa que la presenta
Excepciones	
Frecuencia esperada	Una vez al día
Comentarios	

Tabla 5. Planilla Caso de Uso: Solicitud de envío de datos

Nombre del Caso de Uso	**Envío de datos validados**
Actor Principal	PWS, Proveedor Web Services Empresa Cliente
Versión	1.0
Autor	Angela Reyes

Fecha	Marzo 2004
Actores involucrados	PWS: envía la información que ya ha sido validada, a la empresa que lo solicita Empresa Cliente: recibe los datos del usuario inicial, que ya han sido verificados
Objetivos	Recepción de datos por parte de la empresa cliente
PreCondición	
Secuencia normal	El PWS, envía a la empresa cliente, los datos que ya han sido verificados La empresa cliente, recibe la información solicitada.
Excepciones	
Frecuencia esperada	Una vez al día.
Comentarios	

Tabla 6. Planilla Caso de Uso: Envío de datos validados

4.4.4 DIAGRAMA DE ACTIVIDADES

Debido a que para el diseño de este prototipo se han considerado por separado, el lado del Sistema/ Usuario Inicial, y por otro él módulo, de la empresa cliente; se muestran a continuación los diagramas de actividades, de ambos módulos del prototipo.

4.4.4.1 Diagrama de actividades, módulo Sistema / Usuario Inicial.

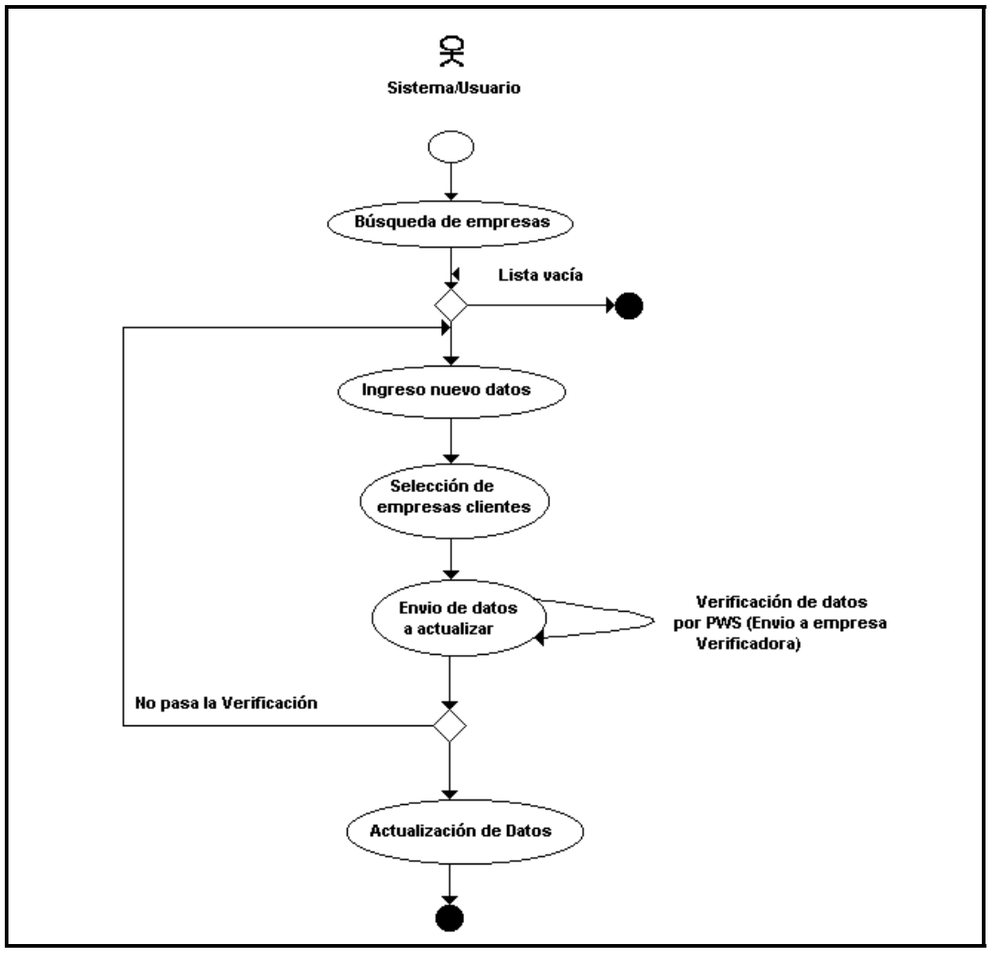


Figura 18, Diagrama de actividades, módulo Sistema / Usuario Inicial.

4.4.4.2 Diagrama de Actividades Módulo Empresa Cliente

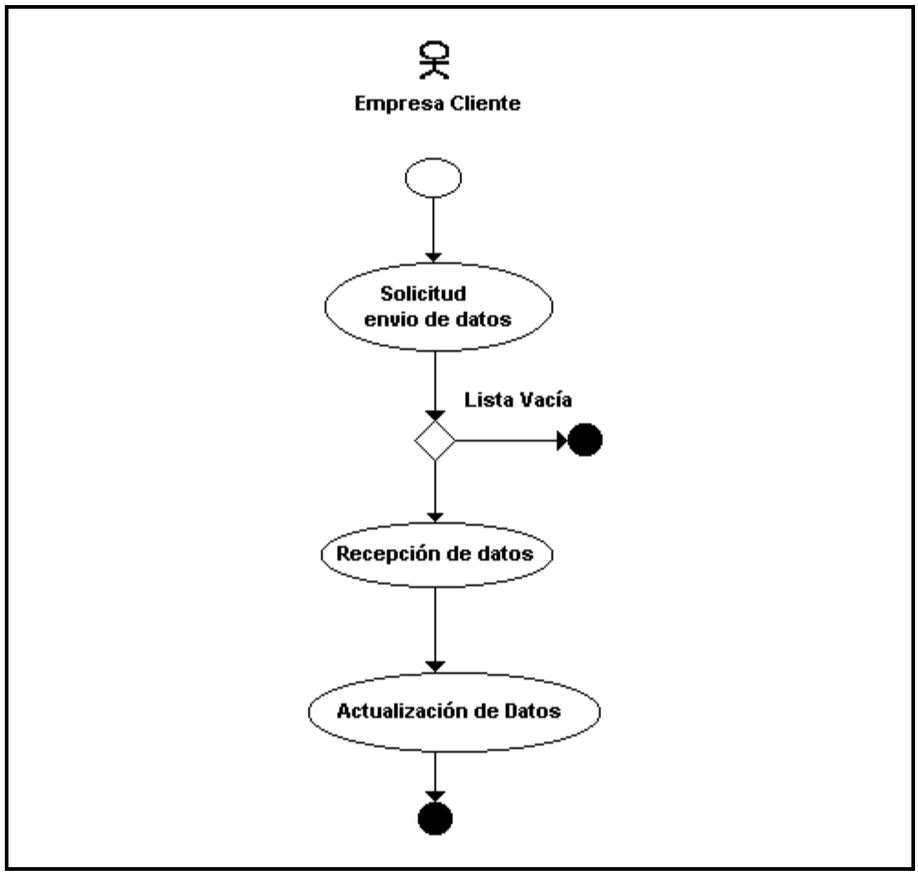


Figura 19, Diagrama de Actividades Módulo Empresa Cliente

4.4.5 DIAGRAMA DE SECUENCIAS

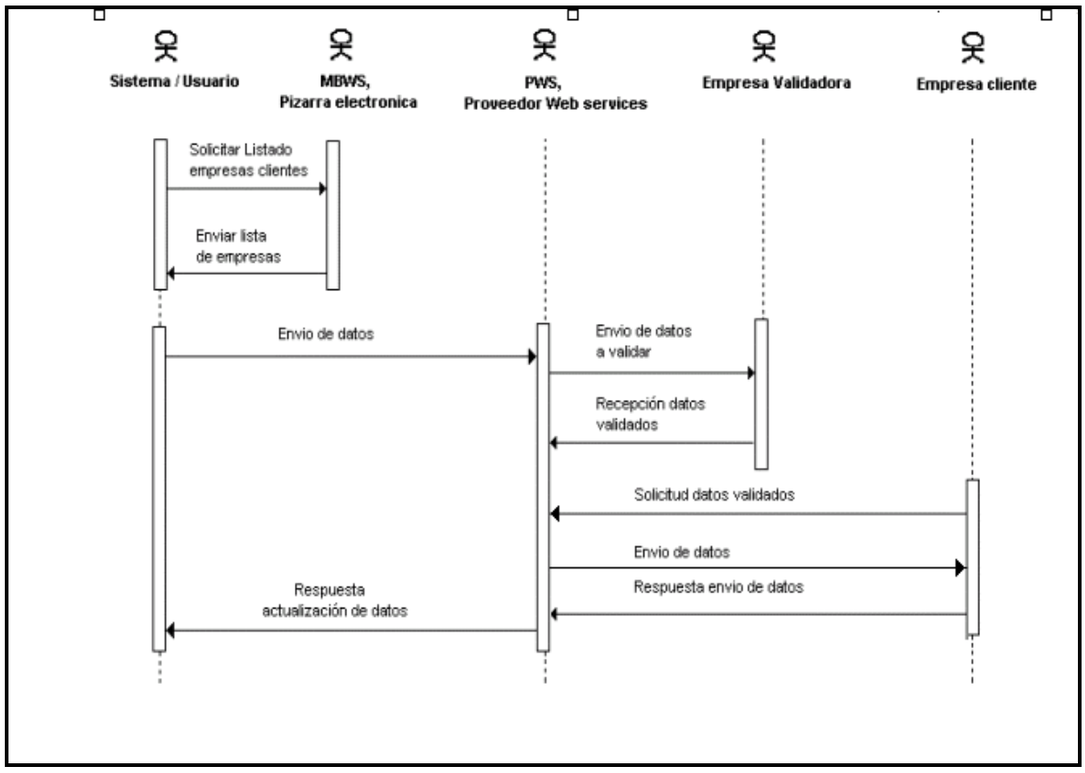


Figura 20, Diagrama de Secuencia

4.5 MODELO DE DATOS

Como parte del diseño de nuestro prototipo, se debe analizar el modelo de datos a utilizar, con el fin de describir los datos a almacenar y el flujo que habrá entre ellos.

Tal como se presenta en el modelo de la arquitectura, y a modo el manejo de los datos se hará sobre bases de datos SQL Server 2000 tanto para la aplicación cliente, como el Web Services. Se hace esta salvedad ya que las bases de datos utilizadas por las aplicaciones clientes, podría ser de cualquier otro tipo.

Como se observa en la siguiente figura, el modelo de datos que utilizará nuestro Web Services será el siguiente:

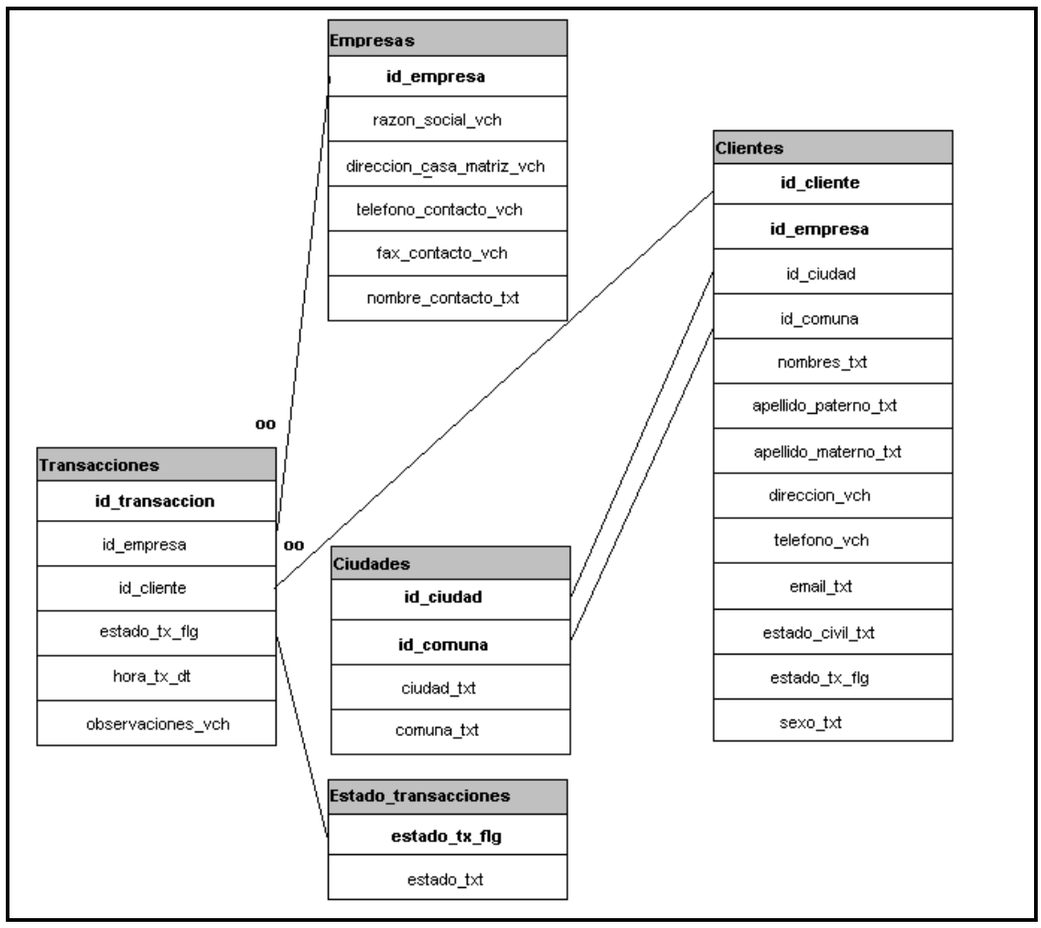


Figura 21, Modelo de Datos Web Services

4.6 DESCRIPCIÓN

A continuación se describen, las tablas y sus respectivos campos, del modelo de datos que utilizará el Web Services

EMPRESAS	**Descripción: Mantiene toda la información de las empresas que tienen contratado el Web Services de actualización de datos.**
Id_empresa	identificador único asociado a cada empresa
Razon_social_vch	Razón social de la empresa
Dirección_casa_matriz_vch	Dirección de la casa matriz de la empresa
Telefono_contacto_vch	Teléfono de contacto de la empresa
Fax_contacto_vch	Fax de contacto de la empresa
Nombre_contacto_txt	Nombre de la persona de contacto en la empresa

Tabla 7. Descripción Modelo de Datos: Tabla empresas

TRANSACCIONES	**Descripción: Mantiene las transacciones realizadas entre las empresas y los clientes**
Id_transacción	identificador único de la transacción
Id_empresa	identificador único de la empresa
Id_cliente	identificador único del cliente
Estado_tx_flg	Estado de la transacción
Hora_tx_dt	Hora de la transacción
Observaciones_vch	Observaciones

Tabla 8. Descripción Modelo de Datos: Tabla Transacciones

CIUDADES	**Descripción: Mantiene el registro de ciudades y comunas del país**
Id_ciudad	identificador único de la ciudad
Id_comuna	identificador único de la comuna

Ciudad_txt	Nombre de la ciudad
Comuna_txt	Nombre de la comuna

Tabla 9. Descripción Modelo de Datos: Tabla Ciudades

ESTADO_TRANSACCIONES	**Descripción: Mantiene los diferentes estados en los que se puede encontrar una transacción**
Estado_tx_flg	identificador único del estado
Estado_txt	Descripción del estado de la transacción

Tabla 10. Descripción Modelo de Datos: Tabla Estado_Transacciones

CLIENTES	**Descripción: Mantiene toda la información de los clientes**
Id_cliente	Identificador único del cliente
Id_empresa	Identificador único de la empresa
Id_ciudad	Identificador único de la ciudad
Id_comuna	Identificador único de la comuna
Nombre_txt	Nombre del cliente
Apellido_paterno_txt	Apellido paterno
Apellido_materno_txt	Apellido materno
Dirección_vch	Dirección del cliente
Telefono_vch	Teléfono del cliente
Email_txt	Email del cliente
Estado_civil_txt	Estado civil
Estado_tx_flg	Estado de la transacción
Sexo_txt	Sexo del cliente

Tabla 11. Descripción Modelo de Datos: Tabla Clientes

4.7 ARQUITECTURA DE COMUNICACIÓN Y TECNOLOGÍAS UTILIZADAS.

En la siguiente figura se observan los diferentes componentes de la arquitectura de comunicación de los Web Services.

Describiremos en este punto, los aspectos centrales de dicho modelo de comunicación.

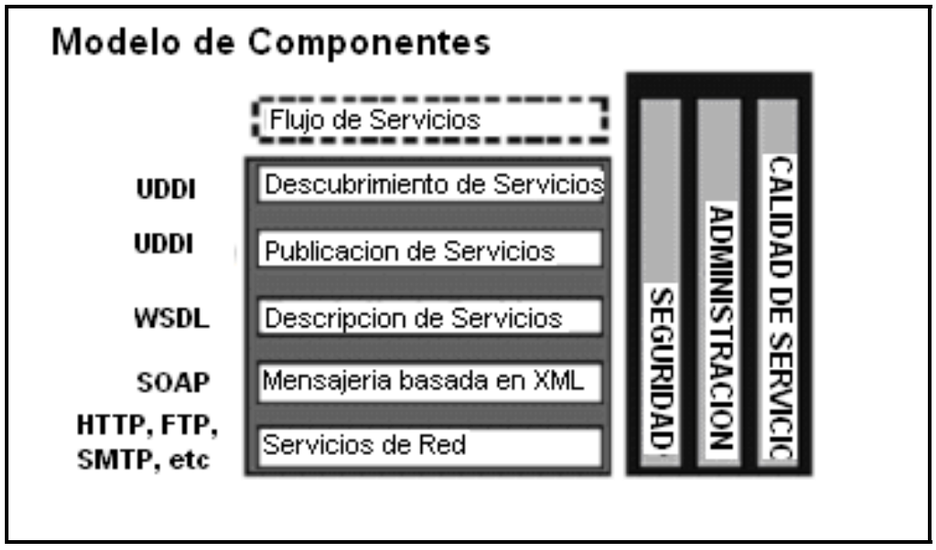


Figura 22, Modelo de componente

Como se vió en el capítulo anterior los servicios web se definen a partir de las siguientes especificaciones:

- SOAP[17]
- WSDL[18]
- UDDI [19]

---

[17] Simple Object Access Protocol

[18] Web Services Description Language

Del mantenimiento de las dos primeras, *SOAP* y *WSDL* se encarga el W3C. En el caso de *UDDI*, se trata de un proyecto en el que participan distintas empresas.

4.7.1 SOAP

Recordando del capítulo 3 lo referente a SOAP; éste corresponde a un protocolo de comunicación que utilizan los Web Services. [URL 1]

La especificación SOAP indica cómo se deben codificar los mensajes que circularán entre las dos aplicaciones, en particular en nuestro desarrollo existirán mensajes SOAP en los siguientes casos:

- Entre el cliente web que se conecta a nuestro Web Services de actualización de datos.
- Entre nuestro Web Services Actualizador de Datos y el "Organismo Validador"

La especificación SOAP define dos modelos de mensajes:

- Un mensaje que se enviará desde la aplicación cliente a la aplicación servidor, solicitando la ejecución de un método al que se pasan una serie de parámetros.
- Un mensaje que se enviará desde la aplicación servidor a la cliente, y que contendrá datos XML con los resultados de la ejecución del método solicitado.

---

[19] Universal Description, Discovery and Integration

4.7.1.1 Ejemplo de Mensaje XML de caso de uso.

A continuación se presenta un ejemplo de un Archivo XML SOAP, de envío de datos a actualizar.

```
<? Xml version = “1.0” enconding =”UTF-8” ?>
<solicitudDeActualizacion>
        <persona>

                <nombreCompleto>
                        <nombre1>Juan </nombre1>
                        <nombre2>Carlos </nombre2>
                        <apellidoPaterno> Perez </apellidoPaterno>
                        <apellidoMaterno> Soto </apellidoMaterno>
                </nombreCompleto>
                <sexo> Masculino </sexo>
                <estadoCivil> Casado </estadoCivil>
                <direccion>Maipu 546</dirrecion>
                <ciudad> Valdivia </ciudad>
                <telefono> 457812</telefono>
                <correoElectronico> jperez@hotmail.com </correoElectronico>
</persona>
<datosEmpresaCliente>
        <razonSocial> Falabella S.A</razonsocial>
        <identificador> 4578 </identificador>
</datosEmpresaCliente>
</solicitudDeActualizacion>
```

Nota: Todos los mensajes XMLs utilizados en la mensajería de nuestro Web Services de actualización de datos y que forman parte de los casos de usos explicados anteriormente, serán incluidos en un Anexo denominado Mensajería SOAP.

4.7.2 WSDL

Permite describir los distintos métodos o funciones que están disponibles en un servicio web, así como su signatura, es decir, el número de argumentos o parámetros que se les debe pasar, y el tipo de dato que devolverá la función como resultado.

Se establece una equivalencia entre el documento WSDL y un “contrato” que especifica los servicios que el servidor se compromete a ofrecer al cliente, siempre que éste los solicite de la forma adecuada.

Los documentos WSDL deben estar disponibles en el servidor web que ofrece los servicios.

Como su creación resulta compleja, las distintas implementaciones de SOAP permiten generar estos archivos de forma sencilla, sin necesidad de conocer los elementos y la estructura del esquema XML en el que se basan.

4.7.3 UDDI

A medida que el número de proveedores de servicios web aumente, será necesario disponer de un sistema de referencia que permita localizar estos servicios. Este es el propósito de este protocolo.

UDDI es un protocolo relacionado con los servicios web y que tiene una grama importancia, si bien no es uno de los componentes básicos de la tecnología sobre la que se construye el paradigma de los servicios web, provee un repositorio centralizado para publicar información técnica acerca de Web Services. [URL 5]

Está compuesto por un conjunto de repositorios y de registradores.

- Los repositorios siguen el modelo de replicación single-master, y propaga los cambios a los demás repositorios. Estos repositorios son en sí también Web Services.
- Un registrador es una empresa que provee servicio de registración para sus clientes – por ejemplo Microsoft tiene un aplicativo que vía una interfase gráfica escrita en HTML, me permite dar de alta en el repositorio.

4.7.4 ARQUITECTURA DE COMUNICACIÓN.

Para analizar con mayor detalle la arquitectura de comunicación utilizada por el servicio Web, se debe observar la siguiente figura:

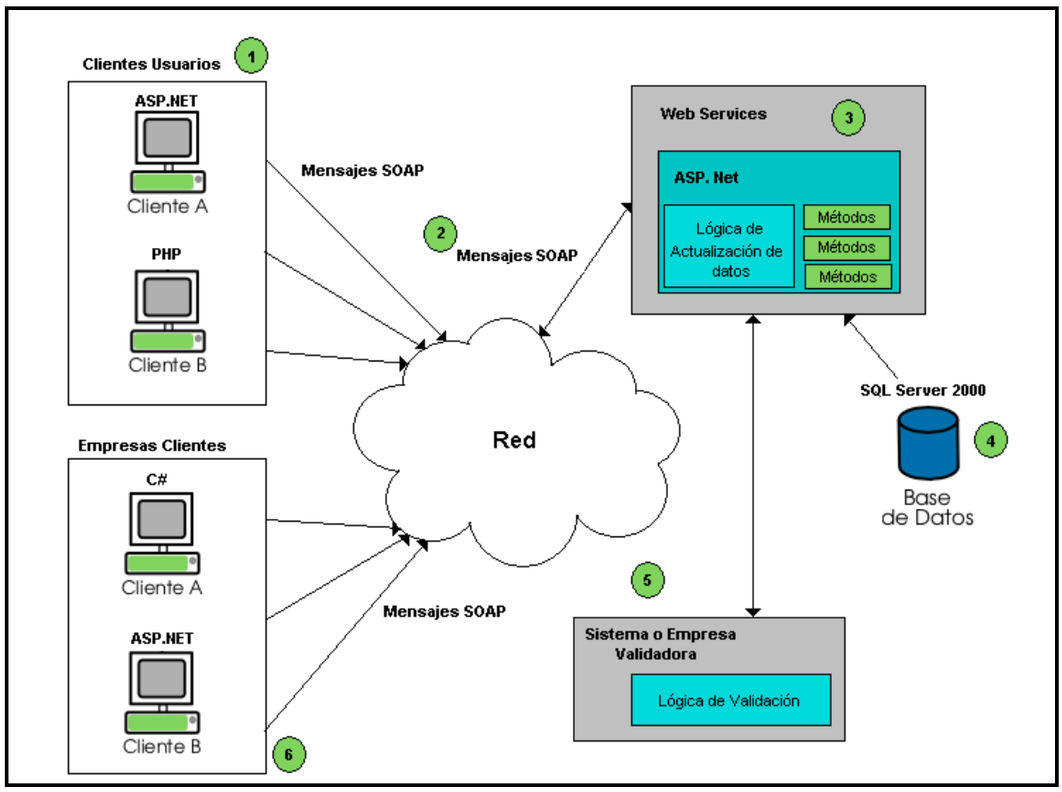


Figura 23, Arquitectura general de comunicación

Como se detalla en la Figura 23, el modelo de la arquitectura de comunicación se basa en el envío de mensajes SOAP a través de la red. Esto es indiferente a la arquitectura utilizada tanto por los clientes como por las empresas.

En particular se describe la arquitectura de comunicación de la siguiente forma.

1. Un GUI[20] escrito en cualquier lenguaje (ASP.NET, C,C++, PHP, JAVA), con librerías para conexión Web Services actuará como cliente usuario y se conectará hacia nuestro Web Services Actualizador de Datos para solicitar cualquier funcionalidad requerida.
   - Inscripción de usuario.
   - Selección de Empresas por Parte del Cliente.

[20] Graphical User Interface

- Administración / Actualización de Datos.

2. Los Mensajes son enviados vía protocolo SOAP (XML) al Web Services de Actualización de Datos. Estos mensajes son dependientes de los servicios o métodos a los cuales se requiera acceder desde la aplicación cliente.

3. Los mensajes SOAP son “procesados” por los distintos métodos o servicios implementados en nuestro Web Services de actualización de datos.

4. Dependiendo del método o servicio que fue solicitado al Web Services, se hace el almacenamiento de la información / transacciones en una base de datos de la empresa que implementa el Web Services.

5. Una vez que se ha ingresado la información necesaria en el Web Services, es enviada al “Organismo Validador” vía Mensajes SOAP. Este organismo procesa la información y entrega o devuelve los datos validados. En particular para la construcción de este prototipo el “Organismo Validador” corresponderá a una simulación interna del sistema.

6. Un GUI escrito en cualquier lenguaje (ASP.NET, C, C++, PHP, JAVA) con librerías para conexión Web Services, actuará como cliente del lado empresa. En particular este cliente solicita los datos actualizados y los ingresa a sus sistemas internos dentro de la empresa.

En resumen el Web Services que se construirá será generado en ASP.NET y la utilización una Base de datos SQL Server 2000, para el almacenamiento y manejo de datos por parte del Web Services.

El flujo de información se realizará desde un cliente que actualizará sus datos y los enviará al Web Services, este los almacenara y enviará a la “entidad validadora”, la cual los devolverá con el resultado del proceso de validación.

De esta forma los datos son actualizados nuevamente en la Base de datos utilizada por el Web Services.

Luego se notificará a la aplicación cliente el resultado de esta transacción, si ésta se ha realizado con éxito, la empresa cliente inscrita podrá solicitar el envío de estos datos para actualizarlos en su base de datos local.

4.7.5 SELECCIÓN DE UN LENGUAJE Y SU ARQUITECTURA PARTICULAR.

En particular existen un sin número de lenguajes de programación como Java, C++, C#, PHP y otros que permiten la implementación de Web Services, es decir proveen de librerías o API's que permiten a los distintos desarrolladores abstraerse de la programación "dura" de los protocolos a nivel de red que maneja SOAP y WSDL.

En este trabajo de titulación se utilizó el lenguaje ASP.NET y su Framework SDK NET debido a la facilidad de uso de éste y a su previo conocimiento y experiencia, sin menoscabar la posibilidad que el prototipo desarrollado en este trabajo, o cualquier otro Web Services pueda ser desarrollado en otro lenguaje.

En particular, a continuación se presenta una breve descripción de algunas características del lenguaje ASP.NET y de su arquitectura para desarrollar Web Services.

4.7.5.1 Por que ASP.NET

En el ámbito de la programación de Servicios Web, se debe considerar que a esta altura, muchos programadores manejan habitualmente para la creación de sus páginas Web, ASP. Si pensamos que asp.net, es la evolución natural del primer lenguaje, resultará entonces muy simple para los programadores acostumbrarse a este nuevo lenguaje. [URL 8]

La creación de Servicios Web utilizando ASP.NET, trae consigo un conjunto de ventajas de programación como las que se mencionan a continuación:

- **Sencillez al crearlos**: Como se ha revisado en la descripción de la mensajería SOAP a lo largo de este documento, podría ser bastante complejo construir toda la base necesaria para llamar y proporcionar los servicios Web (incluyendo crear su propio intérprete de SOAP), pero .NET Framework disminuye esta complejidad, haciendo que los programadores se concentren en el manejo de la reglas de negocio, en vez de los detalles técnicos de enviar y recibir mensajes SOAP.

- **Sencillas al probarlos:** El ambiente .NET ofrece una forma simple de probar fácilmente los Servicios Web que se crean ofreciendo un grupo de pruebas sencillas. Este grupo de pruebas, es también conocido como la página de prueba de los servicios Web o WSDLHelpGenerator.aspx, permite llamar al servicio Web y ver el resultado del método solicitado.

- **Sencillez al implementarlos:** .NET permite tener todas las herramientas para implementar los componentes básicos de un Web Services. Es decir tiene implementados todas las API'a para llamados SOAP, WDSL y UDDI. Lo anterior permite al desarrollador tener un ambiente de programación sencillo y escalable.

4.7.5.2 Funcionamiento de los Servicios Web de ASP.Net

A continuación se describe como funcionan los servicios Web, creados con las herramientas. Net, específicamente con el lenguaje ASP.NET.

Según lo que conocemos hasta el momento, cuando se realiza una solicitud HTTP para un recurso determinado de un servidor Web, por ejemplo una página ASP

Clásica, IIS[21] comprueba su metabase para que la extensión apropiada de **ISAPI** maneje el tipo de archivo solicitado. Para ASP, esa extensión ISAPI es asp.dll. ASP analiza el código, ejecuta sus comandos y devuelve los valores al cliente. Análogamente, cuando se hace una solicitud para un archivo.aspx (Web forms de .Net) o .asmx (Web Services), una extensión de ISAPI llama a IIS para comprobar su metabase y determinar que la extensión ISAPI de ASP.Net, xspisapi.dll, es responsable de manejar esta solicitud. IIS canaliza la solicitud para xspisapi.dll. El xspisapi.dll queda “no administrado”, así que invita al xps.exe, un analizador / controlador “administrativo”, a que asuma el control.

Esto inicia un encadenamiento de llamadas que envía la solicitud HTTP a través de muchas capas de código (módulos HTTP) hasta que se ejecuta el archivo de servicio Web. La siguiente figura ofrece una idea de las capas y secuencias de eventos en el transcurso de la ejecución de un servicio Web.

[21] Internet Information Server

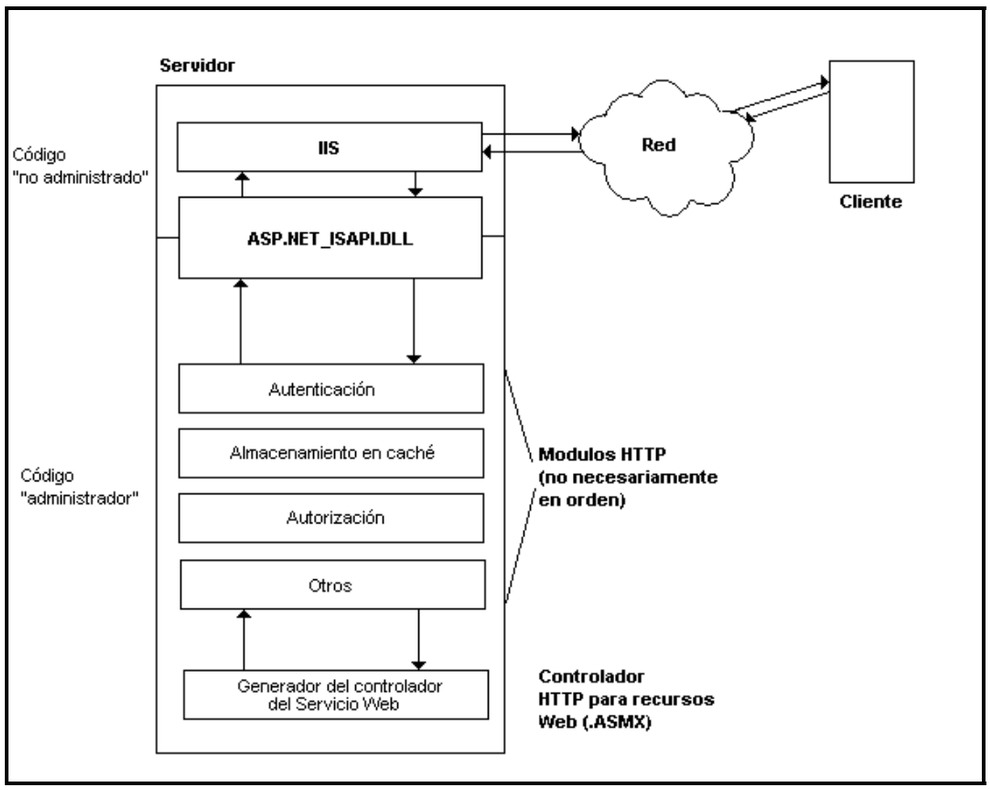


Figura 24, Eventos ejecución Servicio Web

4.8 SERVICIOS O MÉTODOS A IMPLEMENTAR.

Los métodos que se implementaran en el Web Services serán los siguientes:

- **Listado de empresas clientes participantes:** este método será implementado, mediante la creación de una consulta a la base de datos, que retornará los datos principales de las todas empresas clientes participantes.

- **Grabación de datos ingresados:** al invocar este método quedarán almacenados en la base de datos, todos los datos ingresados por el usuario. En un primer momento, es decir, antes de su envío a la empresa verificadora, estos quedarán con estado de transacción 0.

- **Recuperación de datos**: Si el cliente ya ha ingresado sus datos en el sistema, este método permitirá retornar la última información actualizada.

- **Envío de datos a empresa validadora**: este método se llamará de forma interna por el Web Services, y efectuará el envío de los datos a la empresa validadora.

- **Actualización estado de transacción:** este método actualizará el estado, enviado por la empresa verificadora, luego de su validación.

- **Envío de respuesta de actualización**: retornará el estado con el cual retornaron los datos desde el organismo verificador.

- **Listado datos de los clientes, asociado a una empresa determinada:** devolverá los datos resumidos de los clientes que han asociado sus datos a una determinada empresa.

- **Envío de datos a empresa cliente**: este método enviará los datos a la empresa cliente, previa solicitud.

CAPÍTULO 5

5. IMPLEMENTACIÓN Y PRESENTACIÓN DE UN WEB SERVICES

Como se describió en el capítulo anterior, para la buena comprensión de este trabajo se construyó un prototipo de aplicación web que actuará como consumidor cliente de nuestro Web Services de actualización de datos.

Este prototipo será un sitio Web, que constará de dos módulos diferentes. El primero, para los usuarios comunes, el cual se encargará de ser la interfaz gráfica que permita a los usuarios el ingreso y envío de datos hacia nuestro Web Services y por ende consumirá los “servicios” asociados al ingreso y validación de datos. El segundo módulo simulará una interfaz web para las empresas clientes que consumirán los “servicios” asociados a la entrega y actualización de datos de nuestro Web Services.

5.1 FLUJO APLICACIÓN CLIENTE, MÓDULO EMPRESAS

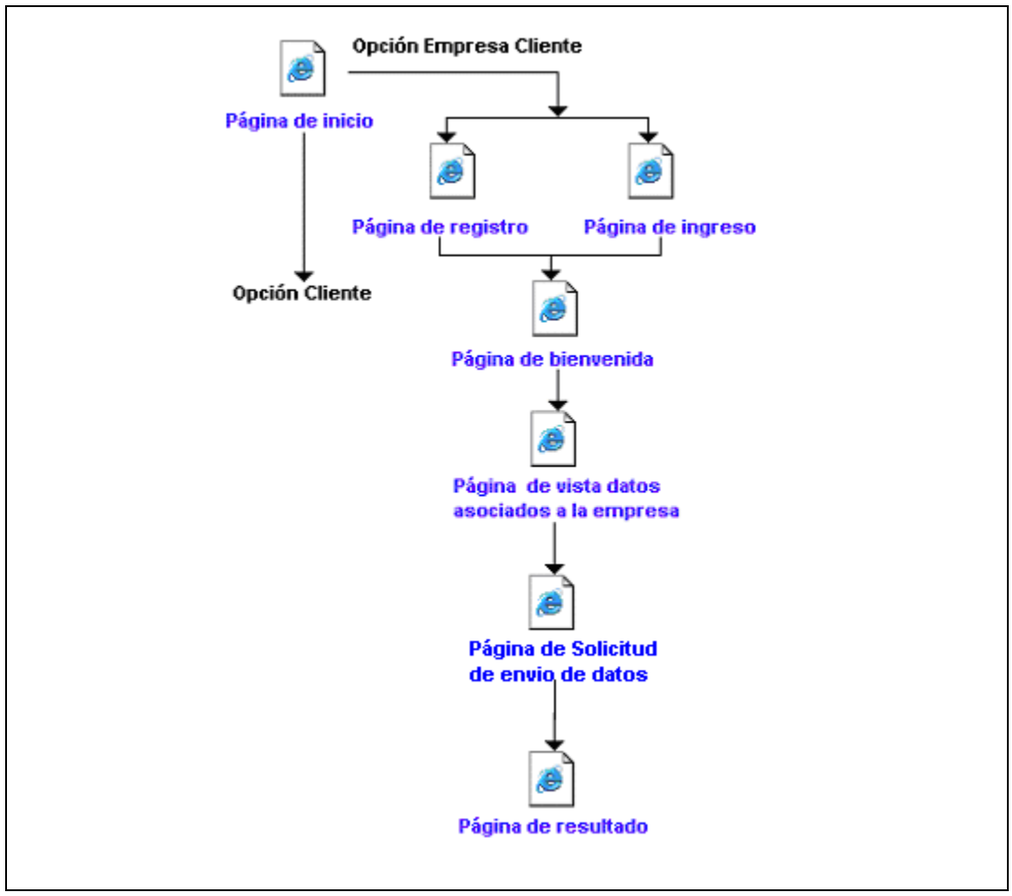


Figura 25, Flujo de Paginas, aplicación Cliente, módulo empresas

En la figura 25, se observa el flujo que tiene la aplicación cliente que consume el Web Services, del lado de la empresa cliente.

Se puede ver que este sitio pequeño web, presenta las funcionalidades descritas en los capítulos anteriores.

1. Validación Ingreso Empresa Cliente.
2. Registro o Verificación de Datos Empresa Cliente.
3. Recepción de datos actualizados / validados de los clientes personas asociados a la Empresa.

Particularmente en la “página de vista de datos asociado a la empresa”, se entrega un listado resumido de los clientes que han solicitado que sus datos sean actualizados para una empresa particular.

5.2 FLUJO APLICACIÓN CLIENTE, MÓDULO PERSONAS

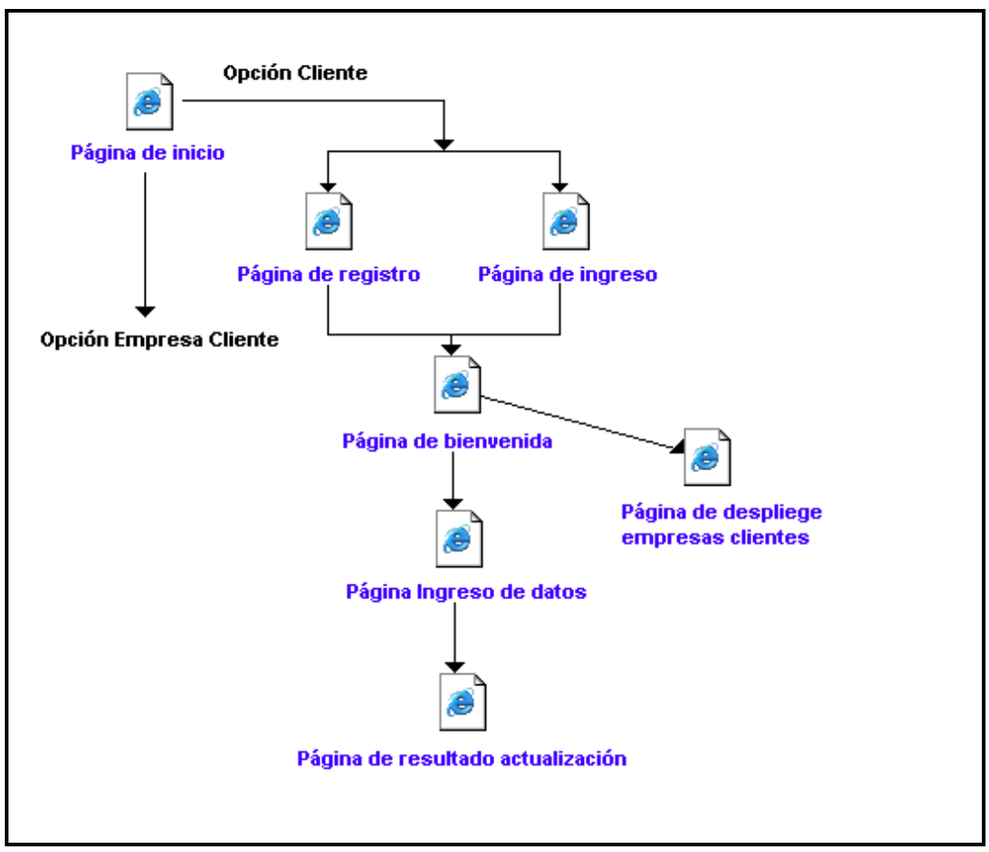


Figura 26, Flujo de Páginas, aplicación Cliente, módulo usuario

Este módulo, entrega la interfaz gráfica que permite al usuario persona, el ingreso y envío de información al Web Services. Como se revisó en el cápitulo 4 las principales funcionalidades que presenta son:

1. Validación Ingreso Usuario Persona.
2. Registro o Verificación de Datos Usuario Persona.
3. Ingreso /Presentación de Datos del Usuario Persona.
4. Presentación de las Empresas asociadas al sistema.

5. Envió de los Datos del Cliente al Web Services para su Validación / actualización.

5.3 PRESENTACIÓN DE LA APLICACIÓN

Para iniciar el funcionamiento de la aplicación cliente, es necesario ejecutar un Browser (Netscape, IExplorer, etc.) y cargar la página asociada al inicio del sistema. En este caso particular las páginas están ubicadas en el servidor intranet de la empresa CREZCA LTDA donde fue realizada la implementación de prueba de este trabajo de titulación.

5.4 INGRESO GENERAL

A continuación se presentan las funcionalidades desarrolladas en el sistema Web, que nos sirve de ejemplo del funcionamiento de nuestro Web Services para la actualización de datos.

5.4.1 PÁGINA DE INICIO

Al iniciar al sistema, se despliega una página de inicio, donde se presentan las diferentes alternativas, para un cliente persona o una empresa.

Figura 27, Página de Inicio, Portal actualización

Como se observa en la figura anterior, las funcionalidades que ofrece esta página, son nada más de enlace a otras, por lo tanto esta pagina, no realiza ninguna llamada a algún método del Web Services

5.5 MÓDULO CLIENTES PERSONAS

5.5.1 INGRESO USUARIO

A continuación se presenta la página de ingreso de un usuario ya existente en el sistema, es decir, que se haya inscrito con anterioridad.

Como se vió, en la página de inicio del sistema se presentaba el enlace a esta página.

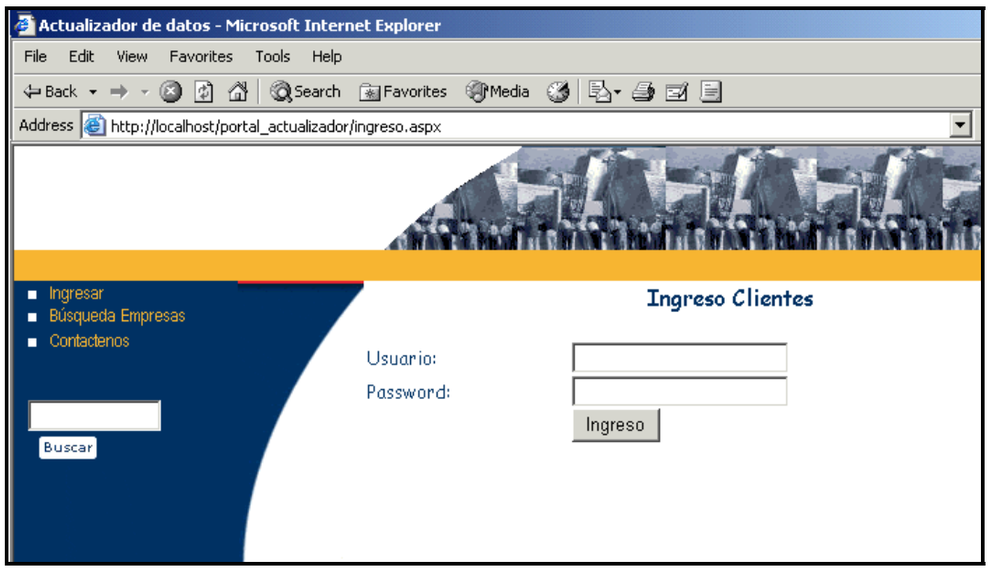


Figura 28, Ingreso Clientes

Es en esta página donde se realizará el control de acceso de los usuarios, para así verificar que realmente estén inscritos al sistema.

La aplicación consumidora del servicio Web, debe cumplir con los estándares de seguridad de este tipo de aplicación.

El control de acceso a la aplicación, se efectúa utilizando un nombre de usuario y un password único para cada usuario. El sistema administra las password considerando:

- Cada usuario tiene una sola password.
- Las passwords no se ven en texto claro, en ninguna instancia donde se utilicen.
- El sistema de autentificación rechaza password con similitudes a las anteriores, según los estándares establecidos.

- El sistema de control de acceso, limita según especificaciones el número de intentos de ingreso con passwords erróneos. Al cumplirse el límite, el usuario es deshabilitado de la utilización del sistema.

Se debe mencionar, como un punto interesante, que al realizar esta aplicación, sobre la plataforma de Microsoft.Net, específicamente al utilizar el lenguaje Asp.net, se comprobaron e implementaron las funcionalidades de autenticación que dicho lenguaje provee.

5.5.2 INGRESO CLIENTES YA EXISTENTES

Al comprobarse que el usuario y contraseña, ingresados por el cliente son correctos, se llama al primer método del Web Services que retornará, todos los datos del cliente que ha ingresado.

Dicho método, realiza la consulta pertinente sobre la tabla “Clientes”, teniendo como condición el identificador del cliente.

En los anexos de este trabajo, se presenta la transacción SOAP XML con la información del usuario utilizado en el método “Recuperación de datos clientes” de nuestro Web Services. Este archivo SOAP XML, podrá ser utilizado por cualquier aplicación que desee manejarlo y que se conecte con nuestro Web Services.

A continuación se presenta la interfaz descrita.

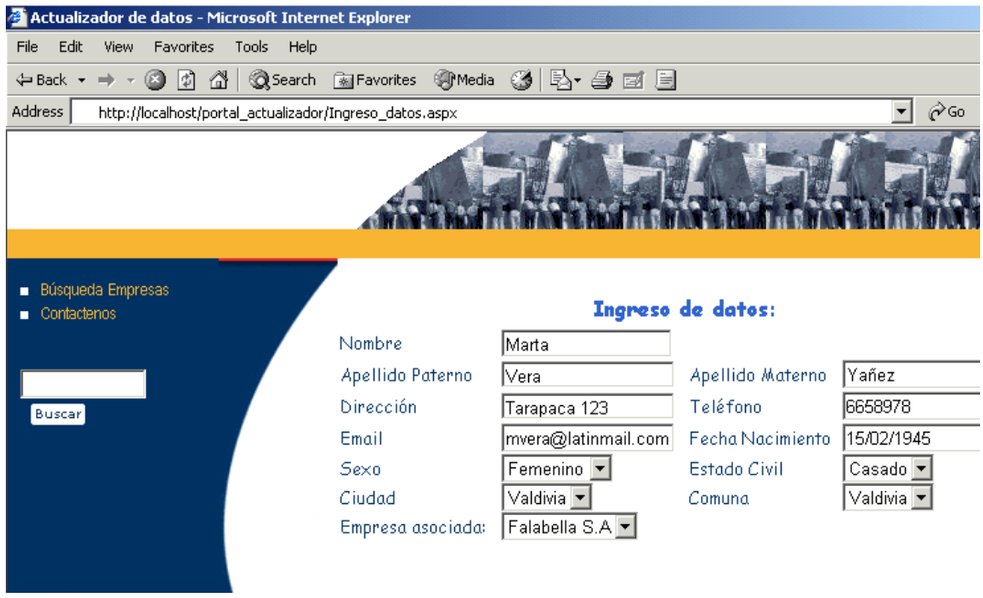


Figura 29, Información cliente

5.5.3 PIZARRA ELECTRÓNICA CON LISTADO DE EMPRESAS

Con esta funcionalidad el usuario tendrá la posibilidad de ver el listado completo de las empresas inscritas en el sistema y de acuerdo a su elección seleccionar a qué empresas él quiere enviar los datos de actualización.

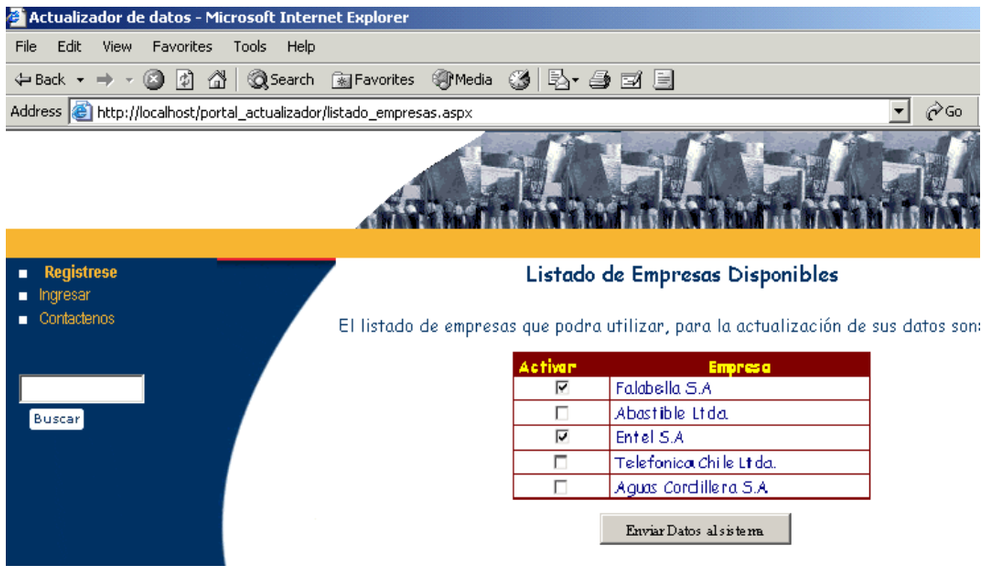


Figura 30, Listado de Empresas y Envío de Información.

Para esto se utiliza un nuevo método implementado, el cual hará la consulta pertinente a la tabla “empresas” y “transacciones” y retornará un archivo SOAP XML, con la información resumida de las empresas asociada al cliente, tal como se observa en la Figura 30.

5.5.4 ENVÍO DE INFORMACIÓN

En el caso de que el cliente sea nuevo, se presentará la misma pantalla vista en la Figura 29. La diferencia radica en la acción que se efectuara al seleccionar el envío de información.

Si el cliente es nuevo, se llamará al método que almacena la información del nuevo usuario en la tabla “Clientes”; si no, se utilizará el método del Web Services que actualiza dicha información.

Al terminar la actualización o ingreso de los datos, el cliente, por medio del botón “Enviar Datos al Sistema”, tendrá la opción de actualizar su información, para posteriormente enviarla a la empresa verificadora.

5.5.5 RESPUESTA A CLIENTE

Una vez que la información ha sido enviada a la empresa actualizadora, el sistema enviará la respuesta al cliente, tal como se ve en la siguiente figura.

Figura 31, Pagina resultado transacción

5.6 MÓDULO EMPRESAS

Como se mencionó, la nuestro sistema web de ejemplo, presentará funcionalidades tanto para clientes personas, así como para empresas.

A continuación se detallarán dichas funcionalidades y métodos consumidos desde el Web Services, en el caso de las empresas.

5.6.1 INGRESO EMPRESA

Para autentificar el ingreso de las empresas clientes de la aplicación, se hará uso de una página similar a la empleada en el caso de los clientes.

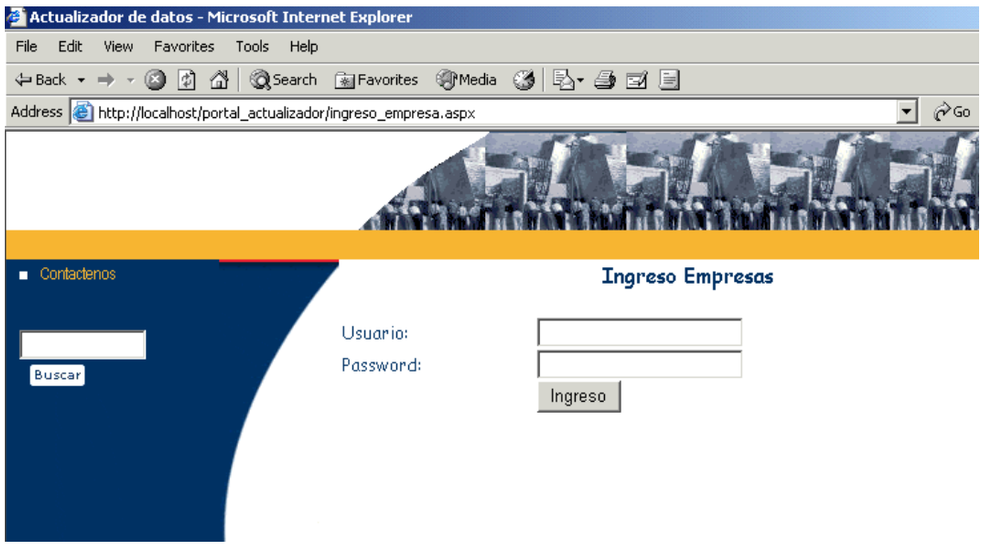


Figura 32, Ingreso empresas

Para esto se utilizará el método del Web Services, que realiza la consulta pertinente sobre la tabla “clientes”, teniendo como condición el identificador de la empresa.

Dicho método, enviará como resultado, un archivo SOAP XML, con toda la información de los clientes asociados a la empresa.

5.6.2 INGRESO NUEVA EMPRESA

Si la empresa no está inscrita en el sistema, se desplegará la página correspondiente a su registro.

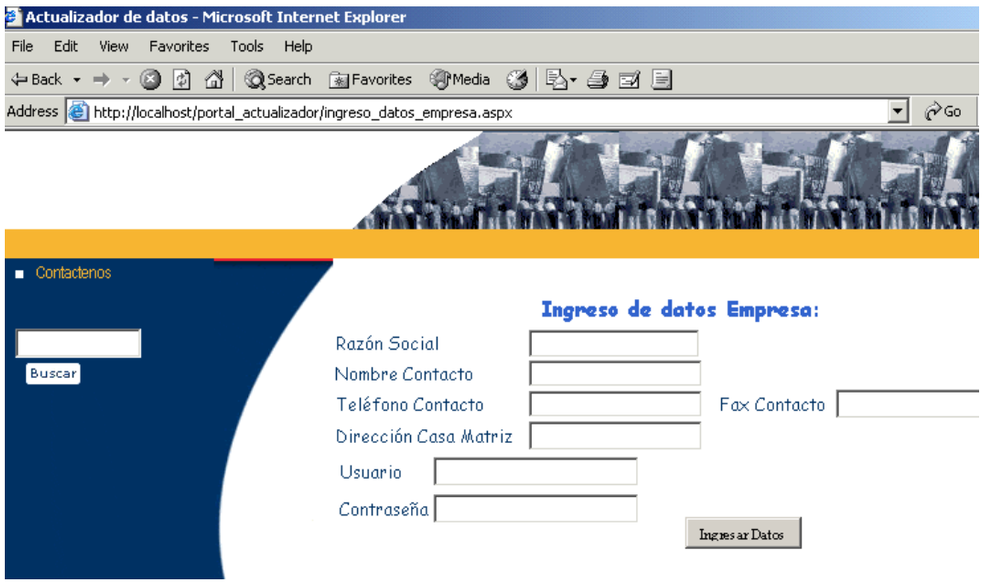


Figura 33, Ingreso Nueva empresa

Para almacenar la información de la empresa en la base de datos, se utilizará un método del Web Services, el cual guardará en la tabla “Empresas”, toda la información previamente ingresada.

5.6.3 INGRESO EMPRESA YA EXISTENTE

Si la empresa que ha ingresado existe en los registros de la base de datos que maneja el Web Services, se desplegarán para ella un listado de los clientes personas que han asociado sus datos a ella (ver figura 34), y por lo tanto cuando estos clientes actualicen sus datos, éstos estarán disponibles para la empresa. Una ves que el administrador o usuario de la interfaz del lado cliente empresa requiera recuperar los datos actualizados de los clientes, hará click en el botón “Recuperar Datos de Clientes Actualizados”, con ello se confirmará su solicitud como se presenta en la figura 35.

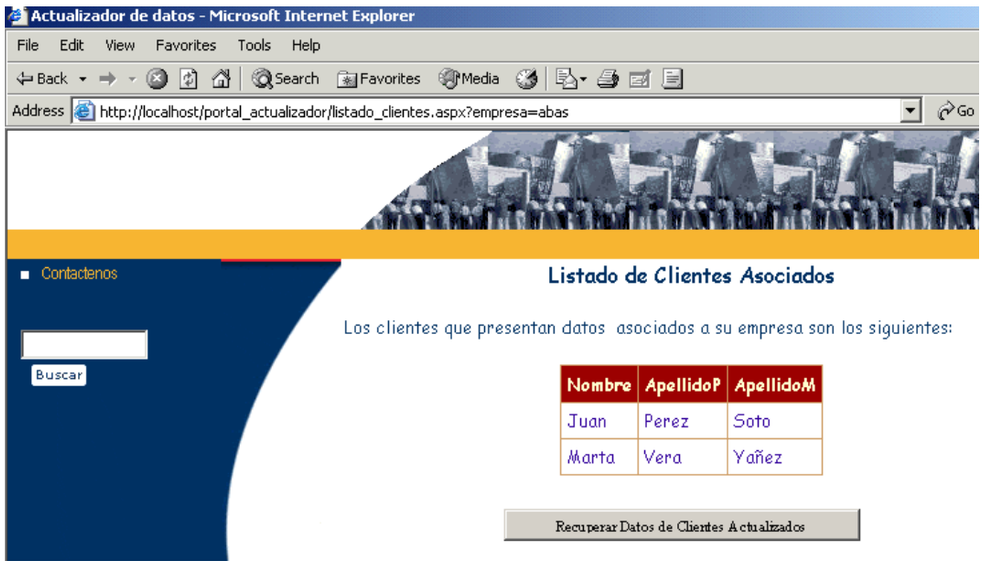


Figura 34, Listado Clientes Asociados

5.6.4 SOLICITUD DE INFORMACIÓN

El envío de la información de clientes personas asociados a la empresa, se hará desde la pantalla que se observa en la siguiente figura.

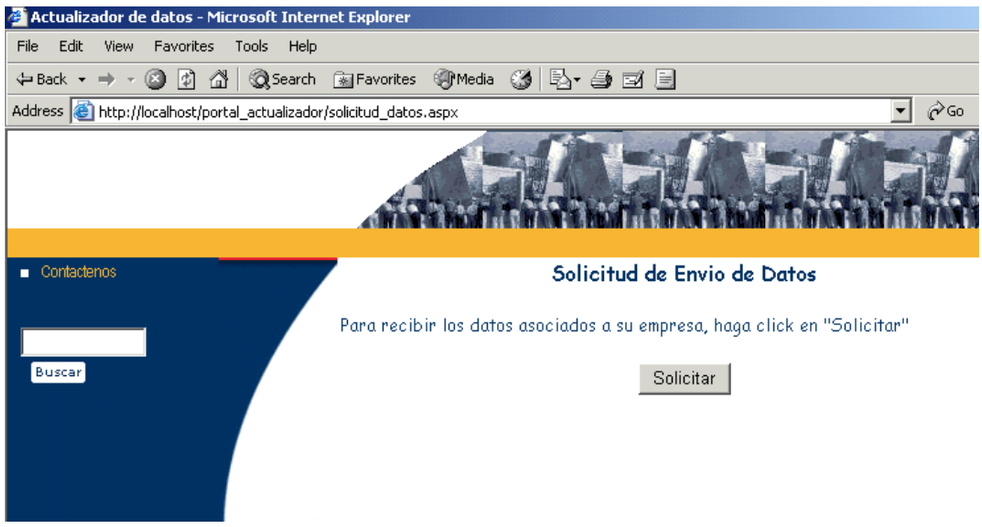


Figura 35, Solicitud transferencia de datos

Con esto la información es enviada a la empresa que lo solicita.

5.6.5 RECEPCIÓN INFORMACIÓN POR PARTE DE LA EMPRESA

El resultado del envío de la información se presenta en la pantalla observada en la siguiente figura.

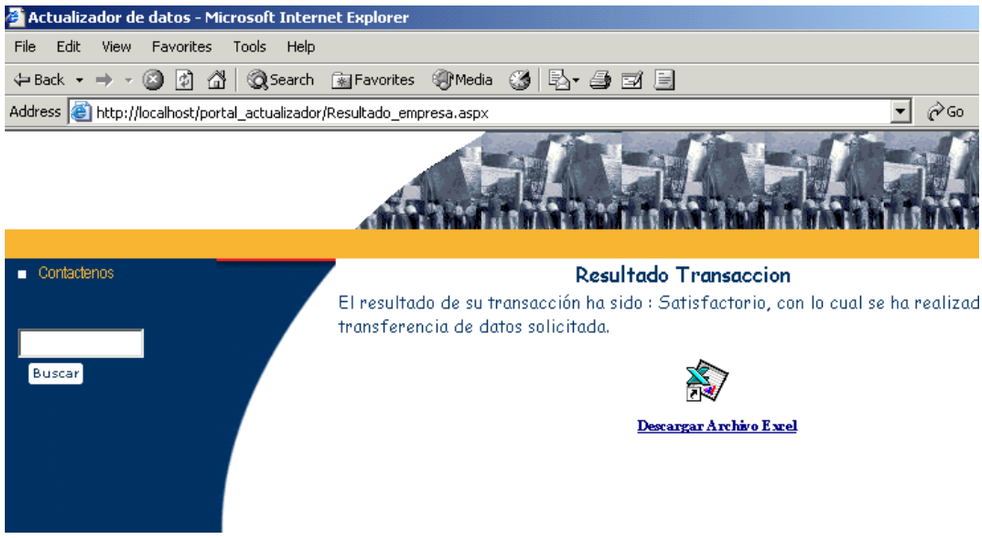


Figura 36, Resultado transacción

En nuestro caso y a modo de ejemplo se recuperó la información de los clientes actualizados y es posible descargarla en formato Excel para el uso que se estime conveniente. En un caso real, la empresa podría ingresar directamente los datos actualizados a sus sistemas internos de facturación, logística, etc.

Como se ha podido observar, los métodos implementados en el Web Services, podrán ser llamados desde cualquier tipo de aplicación, tanto de escritorio, como una aplicación Web.

Además en el caso de que se desee en el futuro, implementar algún otro método sobre el servicio web, bastará que el administrador de la aplicación cliente, actualice la referencia a éste, para poder hacer uso de este nuevo método.

CAPÍTULO 6

6. CONCLUSIONES

Actualmente las empresas utilizan Internet como un medio donde concretar una serie de transacciones comerciales.

Desde este punto de vista Internet ha evolucionado considerablemente, si recordamos sus inicios, donde simplemente era utilizado como medio de consulta de información y vemos su presente, donde el e-bussines se ha transformado en una de sus aristas mayormente utilizadas, por todas aquella empresas que necesitan realizar sus transacciones con clientes en cualquier lugar geográfico, ahorrando así tiempo y todos los trámites burocráticos de algunas de sus transacciones. Esto ha significado que cada vez más sean los clientes que utilicen este medio para realizar desde compras en línea, a diferentes tipos de pago y otras transacciones.

Dentro de este escenario la incorporación de tecnología distribuida, ofrece un avance considerable en los tipos de servicios que las diferentes empresas puedan prestar.

Como se analizó e investigó en este trabajo de tesis, uno de los modelos que empieza a ser masivamente considerado por los desarrolladores es el de Web Services, el cual gracias a sus considerables ventajas esta siendo ampliamente utilizado, para la realización de todo tipo de transacciones.

Dentro de las ventajas que fueron analizadas durante el desarrollo de este trabajo, destaca su modularidad y su independencia de plataformas y lenguajes, con esto se

permite a las empresas consumir este tipo de servicio independiente del lenguaje que utilicen en sus transacciones comerciales.

Como se sabe, además de considerar como transacciones comerciales, al flujo de información de compras o ventas realizadas sobre Internet, se debe entender que este concepto se puede ampliar también a todo tipo de flujo de información entre una empresa proveedora de un servicio y un cliente que requiere y solicita dicho servicio. Es por esto que la actualización de la información personal de los clientes en los registros de una empresa, es también un tipo de transacción comercial.

El concepto general que se desarrolló en este trabajo de titulación, fue la investigación y creación de un prototipo de Web Services, que permite a los usuarios actualizar su información personal y permitir con esto que diferentes empresas actualicen dicha información en sus propias sistemas internos.

Este tipo de servicio al ser construido sobre la base del modelo de Web Services permitirá su utilización por cualquier tipo de aplicación, permitiendo así que diferentes entidades lo utilicen para el propósito que consideren pertinente.

Como se ha visto la utilización de este tipo de tecnología distribuida, así como los protocolos y lenguajes que la acompañan, como SOAP, WSDL, XML, etc., están permitiendo que la utilización de Internet se amplíe a conceptos que hasta hace un tiempo eran totalmente desconocidos, y brindando así que el concepto de e-commerce esté al alcance de cualquier compañía que desee hacer uso de servicios como los implementados en este trabajo de titulación.

6.1 MEJORAS

- Hacer una investigación más exhaustiva de los distintos lenguajes y sus ventajas y desventajas en el ámbito de los Web Services. Lo cual permitirá tener una visión más amplia al momento de implementar una solución Web Services.

- Incluir el tema seguridad, es decir encriptación entre la mensajería SOAP que viaja entre los participantes en los Web Services.

- Mejoras en las funcionalidades del prototipo, pues este correspondió a un ejemplo básico de implementación de un Web Services, y obviamente se puede mejorar.

BIBLIOGRAFÍA

6.2 DOCUMENTOS

- [Amo00] Amor, D. (2000). *La (R) Evolución del E-Business*. Buenos Aires. Prentice Hall

- [Bri02] Brickley D. , Guha R. (2002) ,*Resource Description Framework (RDF) Model and Syntax Specication. W3C Recommendation* http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/.

- [Cab02] Cabreraetal F. (2002).*Web Services Coordination (WS - Coordination).* http://www.ibm.com/developerworks/library/ws-coor/

- [Chr01] Christensen E., Curbera F., Meredith G., Weerawarana S. (2001), *Web Services Description Language (WSDL)* http://www.w3.org/TR/2001/NOTE-wsdl-20010315.

- [Cur01] Curbera F., Goland Y., Klein J., Leymann F., Roller D., Thatte S., Weer-Awarana S. (2001). *Business Process Execution Language for Web Services* http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/.

- [Ano01] Anónimo.(2001).*Distributed Transaction Processing: The XA Specification,*
- *X/Open Snapshot*

- [Eva98] Evans, K., Klein J.,Lyon J. (1998). *Transaction Internet Protocol Version 3.0. IETF RFC 2371*. http://www.ietf.org/rfc/rfc2371.txt

- [Gov01]Govindarajan K., Karp A., Kuno H., Beringer D.,Banerji A. (2001), *Conversation Definitions: defining interfaces of web services*

- [Gra92]Gray J. N., Reuter A. (1992). *Transaction Processing: Concepts and Techniques*. San Francisco: Morgan Kaufmann.

- [Liz02]Lizarraga Celaya, Carlos.(2002). *Servicios Web*. Universidad de Sonora.

- [Rio02] Riordan, Rebecca. (2002). *Microsoft ADO.NET*. Madrid. McGraw-Hill.

- [Rei02] Reilly, Douglas.(2002). *Microsoft ADO.NET*. Madrid. McGraw-Hill.

- [Tab02] Tabor, R.(2002). *Servicios Web XML*. España. Prentice Hall

6.3 DIRECCIONES EN INTERNET

Direcciones destacadas, ordenadas por tema.

- **Especificación de SOAP**

  [URL 1] http://www.w3.org/TR/SOAP

- **Especificación de WSDL**

  [URL 2] http://www.w3.org/TR/wsdl

- **Microsoft XML Web Services.**

  [URL 3]

  http://msdn.microsoft.com/library/default.asp?url=/nhp/Default.asp?conten

  tid=28000442

- **.Net**

  [URL 4]

  http://www.microsoft.com/latam/net/introduccion/webservices.asp

- **UDDI**

  [URL 5] http://www.uddi.org

  [URL 6] http://uddi.microsoft.com

- **Web Services**

  [URL 7] www.webservices.org

  [URL 8] http://www.411asp.net

  [URL 9] http://www.salcentral.com

- **XML**

  [URL 10] http://www.w3.org/TR/REC-xml

ANEXOS

7. ANEXOS CODIGOS SOAP XML

Se anexan a este trabajo algunos códigos ejemplos de los Mensajes SOAP XML de transacciones en el sistema implementado.

- **Listado de empresas clientes participantes**

```
<?xml version="1.0" encoding="utf-8" ?>
- <DataSet xmlns="http://tempuri.org/">
  - <xs:schema id="NewDataSet" xmlns=""
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:msdata="urn:schemas-microsoft-com:xml-
      msdata">
    - <xs:element name="NewDataSet"
        msdata:IsDataSet="true" msdata:Locale="es-ES">
      - <xs:complexType>
        - <xs:choice maxOccurs="unbounded">
          - <xs:element name="Empresas">
            - <xs:complexType>
              - <xs:sequence>
                  <xs:element name="Codigo"
                    type="xs:int"
                    minOccurs="0" />
                  <xs:element name="Nombre"
                    type="xs:string"
                    minOccurs="0" />
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:choice>
        </xs:complexType>
      </xs:element>
    </xs:schema>
  - <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-
      com:xml-msdata" xmlns:diffgr="urn:schemas-microsoft-
      com:xml-diffgram-v1">
    - <NewDataSet xmlns="">
      - <Empresas diffgr:id="Empresas1"
          msdata:rowOrder="0">
          <Codigo>1</Codigo>
          <Nombre>Falabella S.A</Nombre>
        </Empresas>
      - <Empresas diffgr:id="Empresas2"
          msdata:rowOrder="1">
```

```
            <Codigo>2</Codigo>
            <Nombre>Abastible Ltda.</Nombre>
          </Empresas>
        </NewDataSet>
      </diffgr:diffgram>
    </DataSet>
```

- **Grabación de datos ingresados**

```
<?xml version="1.0" encoding="utf-8" ?>
<boolean xmlns="http://tempuri.org/">true</boolean>
```

- **Recuperación de datos clientes**

```
<?xml version="1.0" encoding="utf-8" ?>
- <DataSet xmlns="http://tempuri.org/">
  - <xs:schema id="NewDataSet" xmlns=""
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:msdata="urn:schemas-microsoft-com:xml-
      msdata">
    - <xs:element name="NewDataSet"
        msdata:IsDataSet="true" msdata:Locale="es-ES">
      - <xs:complexType>
        - <xs:choice maxOccurs="unbounded">
          - <xs:element name="clientes">
            - <xs:complexType>
              - <xs:sequence>
                  <xs:element name="Nombre"
                    type="xs:string"
                    minOccurs="0" />
                  <xs:element
                    name="ApellidoP"
                    type="xs:string"
                    minOccurs="0" />
                  <xs:element
                    name="ApellidoM"
                    type="xs:string"
                    minOccurs="0" />
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:choice>
        </xs:complexType>
      </xs:element>
    </xs:schema>
  - <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-
      com:xml-msdata" xmlns:diffgr="urn:schemas-microsoft-
      com:xml-diffgram-v1">
    - <NewDataSet xmlns="">
      - <clientes diffgr:id="clientes1"
          msdata:rowOrder="0">
```

<Nombre>**Juan**</Nombre>
<ApellidoP>**Perez**</ApellidoP>
<ApellidoM>**Soto**</ApellidoM>
</clientes>
- <clientes diffgr:id="**clientes2**"
msdata:rowOrder="**1**">
<Nombre>**Marta**</Nombre>
<ApellidoP>**Vera**</ApellidoP>
<ApellidoM>**Yañez**</ApellidoM>
</clientes>
</NewDataSet>
</diffgr:diffgram>
</DataSet>

- **Recuperación de un cliente especifico**

<?xml version="1.0" encoding="utf-8" ?>
- <DataSet xmlns="**http://tempuri.org/**">
- <xs:schema id="**NewDataSet**" xmlns=""
xmlns:xs="**http://www.w3.org/2001/XMLSchema**"
xmlns:msdata="**urn:schemas-microsoft-com:xml-msdata**">
- <xs:element name="**NewDataSet**"
msdata:IsDataSet="**true**" msdata:Locale="**es-ES**">
- <xs:complexType>
- <xs:choice maxOccurs="**unbounded**">
- <xs:element name="**clientes**">
- <xs:complexType>
- <xs:sequence>
<xs:element
name="**id_cliente**"
type="**xs:int**"
minOccurs="**0**" />
<xs:element
name="**nombre_txt**"
type="**xs:string**"
minOccurs="**0**" />
<xs:element
name="**apellido_paterno_t**
xt" type="**xs:string**"
minOccurs="**0**" />
<xs:element
name="**apellido_materno_**
txt" type="**xs:string**"
minOccurs="**0**" />
<xs:element
name="**direccion_vch**"
type="**xs:string**"
minOccurs="**0**" />
<xs:element
name="**telefono_vch**"

```
                    type="xs:string"
                    minOccurs="0" />
                <xs:element
                    name="email_txt"
                    type="xs:string"
                    minOccurs="0" />
                <xs:element
                    name="fecha_nacimiento_
                    dt" type="xs:dateTime"
                    minOccurs="0" />
                <xs:element name="sexo_txt"
                    type="xs:string"
                    minOccurs="0" />
                <xs:element
                    name="estado_civil_txt"
                    type="xs:string"
                    minOccurs="0" />
                <xs:element
                    name="id_ciudad"
                    type="xs:short"
                    minOccurs="0" />
                <xs:element
                    name="id_empresa"
                    type="xs:int"
                    minOccurs="0" />
                <xs:element
                    name="estado_tx_flg"
                    type="xs:int"
                    minOccurs="0" />
                <xs:element
                    name="id_comuna"
                    type="xs:short"
                    minOccurs="0" />
                <xs:element name="login"
                    type="xs:string"
                    minOccurs="0" />
                <xs:element
                    name="password"
                    type="xs:string"
                    minOccurs="0" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:choice>
</xs:complexType>
</xs:element>
</xs:schema>
- <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-
    com:xml-msdata" xmlns:diffgr="urn:schemas-microsoft-
    com:xml-diffgram-v1">
  - <NewDataSet xmlns="">
```

```
- <clientes diffgr:id="clientes1"
      msdata:rowOrder="0">
      <id_cliente>18</id_cliente>
      <nombre_txt>Marta</nombre_txt>

        <apellido_paterno_txt>Vera</apellido_paterno
        _txt>

        <apellido_materno_txt>Yañez</apellido_mate
        rno_txt>
      <direccion_vch>Tarapaca 123</direccion_vch>
      <telefono_vch>6658978</telefono_vch>
      <email_txt>mvera@latinmail.com</email_txt>
      <fecha_nacimiento_dt>1945-02-
        15T00:00:00.0000000-
        04:00</fecha_nacimiento_dt>
      <sexo_txt>F</sexo_txt>
      <estado_civil_txt>S</estado_civil_txt>
      <id_ciudad>1</id_ciudad>
      <id_empresa>2</id_empresa>
      <estado_tx_flg>0</estado_tx_flg>
      <id_comuna>2</id_comuna>
      <login>temp</login>
      <password>temp</password>
    </clientes>
  </NewDataSet>
 </diffgr:diffgram>
</DataSet>
```


- **Actualización estado de transacción**


```
<?xml version="1.0" encoding="utf-8" ?>
<boolean xmlns="http://tempuri.org/">true</boolean>
```


- **Listado datos del clientes, asociado a una empresa determinada:**


```
<?xml version="1.0" encoding="utf-8" ?>
- <DataSet xmlns="http://tempuri.org/">
  - <xs:schema id="NewDataSet" xmlns=""
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:msdata="urn:schemas-microsoft-com:xml-
      msdata">
    - <xs:element name="NewDataSet"
        msdata:IsDataSet="true" msdata:Locale="es-ES">
      - <xs:complexType>
        - <xs:choice maxOccurs="unbounded">
          - <xs:element name="clientes">
            - <xs:complexType>
              - <xs:sequence>
```

```
                    <xs:element name="Nombre"
                      type="xs:string"
                      minOccurs="0" />
                    <xs:element
                      name="ApellidoP"
                      type="xs:string"
                      minOccurs="0" />
                    <xs:element
                      name="ApellidoM"
                      type="xs:string"
                      minOccurs="0" />
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:choice>
          </xs:complexType>
        </xs:element>
      </xs:schema>
    - <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-
        com:xml-msdata" xmlns:diffgr="urn:schemas-microsoft-
        com:xml-diffgram-v1">
      - <NewDataSet xmlns="">
        - <clientes diffgr:id="clientes1"
            msdata:rowOrder="0">
            <Nombre>Juan</Nombre>
            <ApellidoP>Perez</ApellidoP>
            <ApellidoM>Soto</ApellidoM>
          </clientes>
        - <clientes diffgr:id="clientes2"
            msdata:rowOrder="1">
            <Nombre>Marta</Nombre>
            <ApellidoP>Vera</ApellidoP>
            <ApellidoM>Yañez</ApellidoM>
          </clientes>
        </NewDataSet>
      </diffgr:diffgram>
    </DataSet>
```